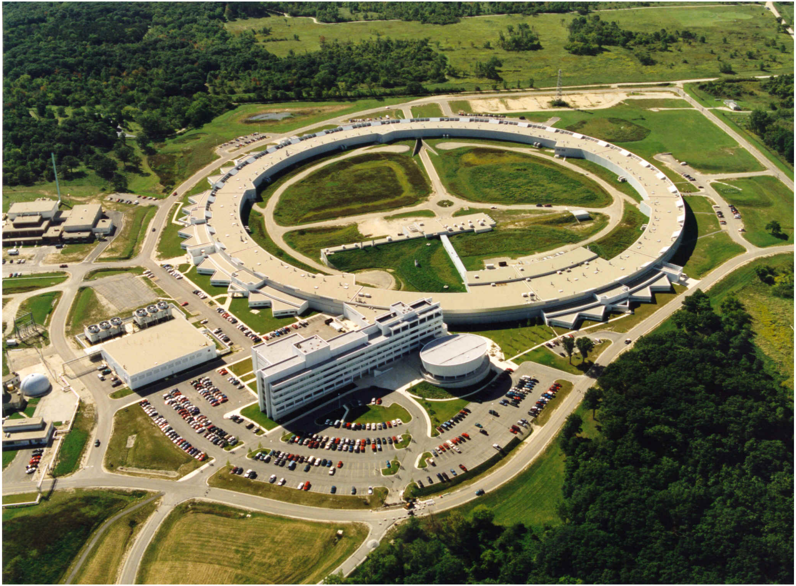
Designing Particle Accelerators Using Derivative-free Optimization

Jeffrey Larson

Argonne National Laboratory

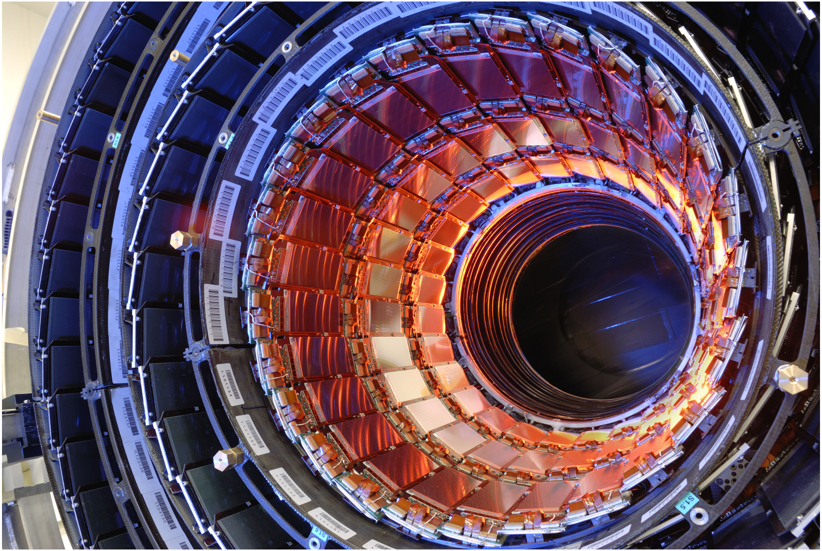
February 6, 2015

Motivation



APS: $467M to build; tens of millions to operate.

Motivation

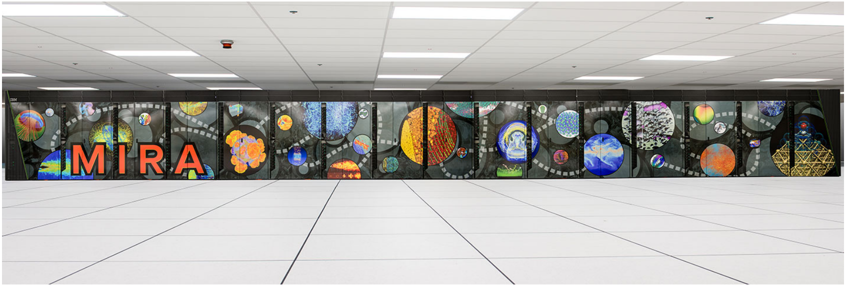


LHC: \$4.75B to build; \$1B to operate; \$0.25B computation

Motivation

Mira: $180M to build; $4M to operate; 3.9 Megawatts

Motivation



Optimizing with gradients

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

- Classical Problem

Optimizing with gradients

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

- Classical Problem

- Derivatives are traditionally very important in optimization

Optimizing with gradients

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

- Classical Problem
- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?

Optimizing with gradients

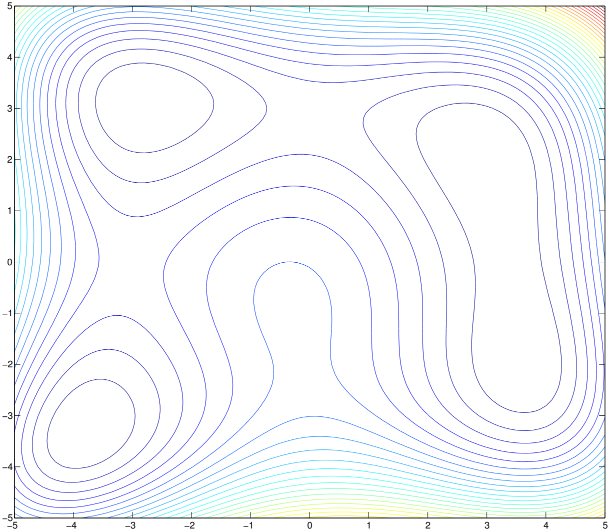
$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \le x \le u < \infty\}$$

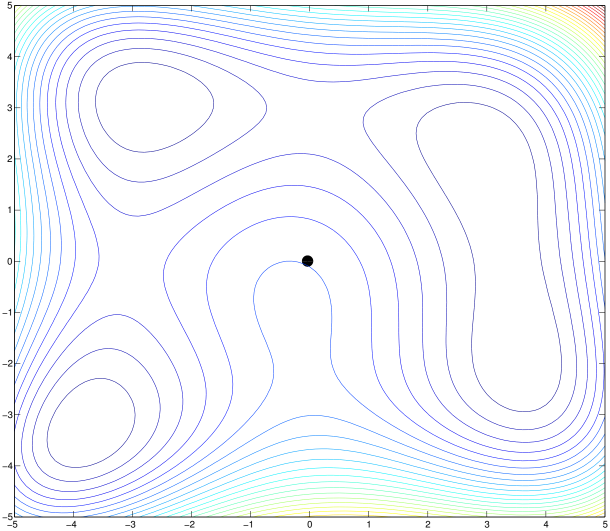
- Classical Problem

- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?
  - What information is contained in the gradient?

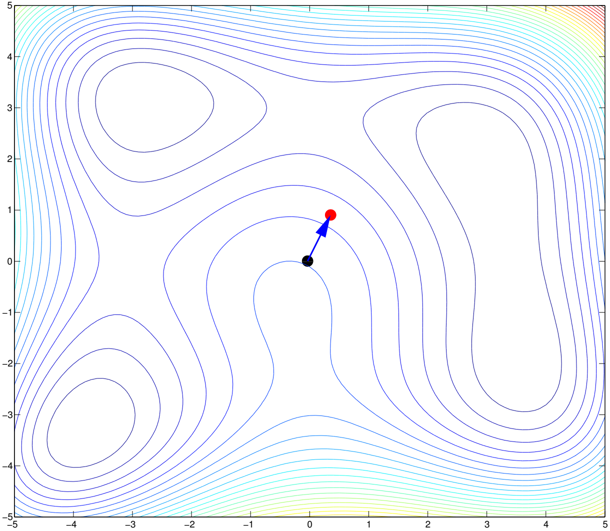
Line Search



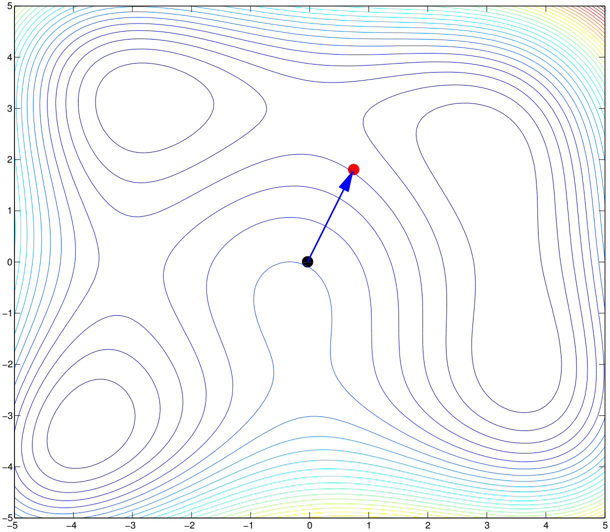
Line Search



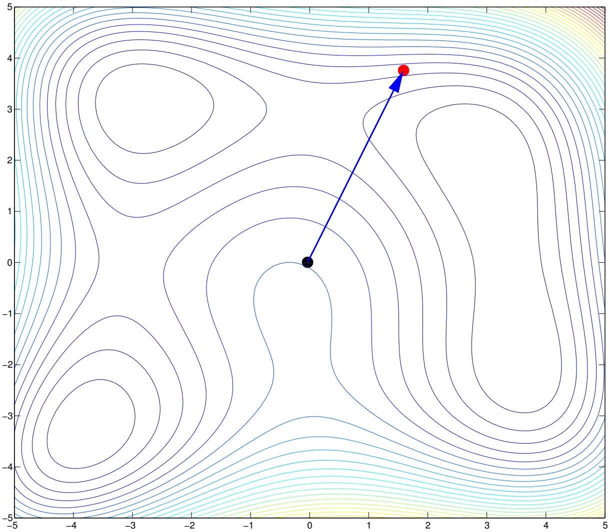
Line Search



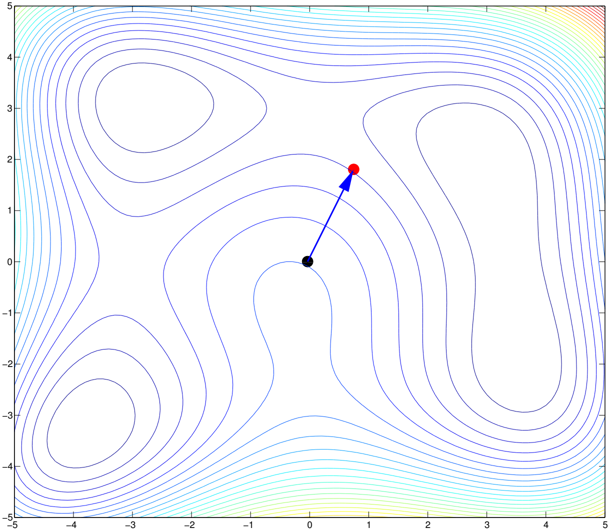
Line Search



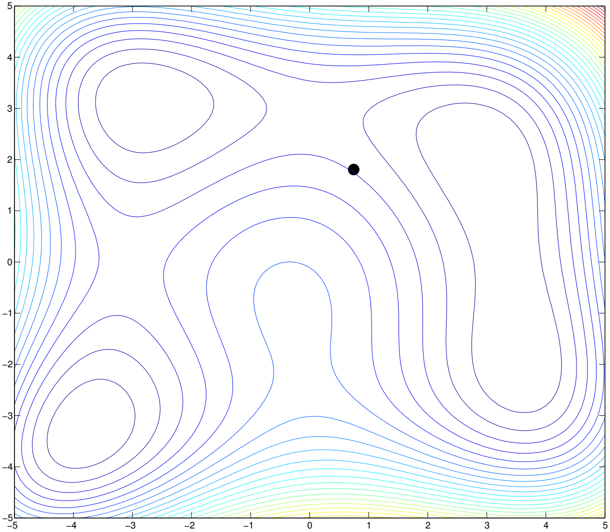
Line Search



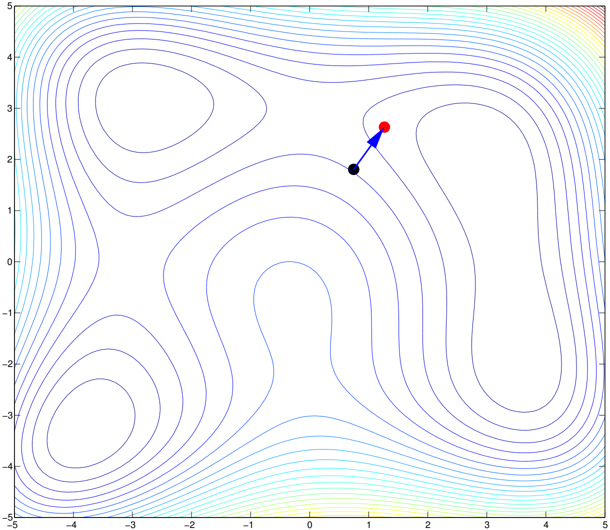
Line Search



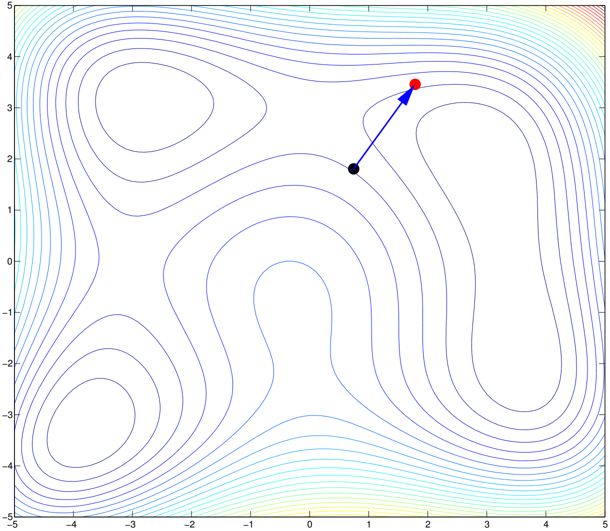
Line Search



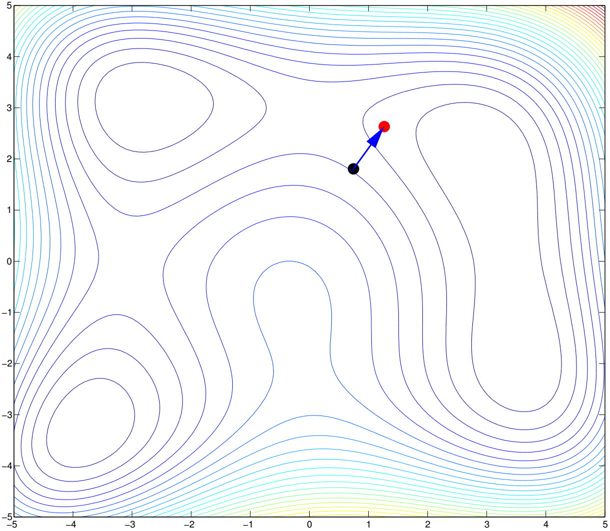
Line Search



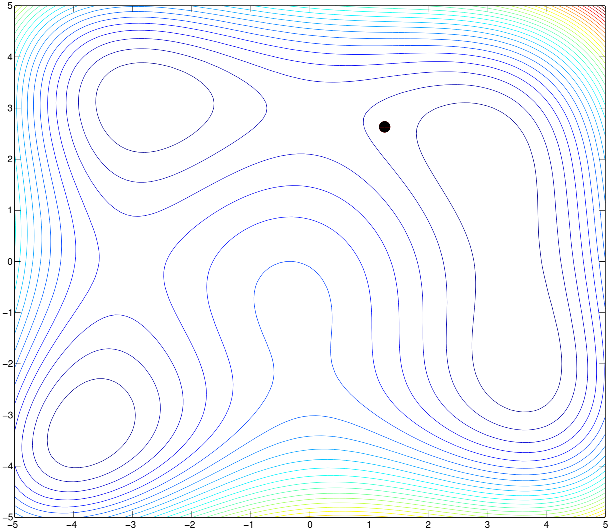
Line Search



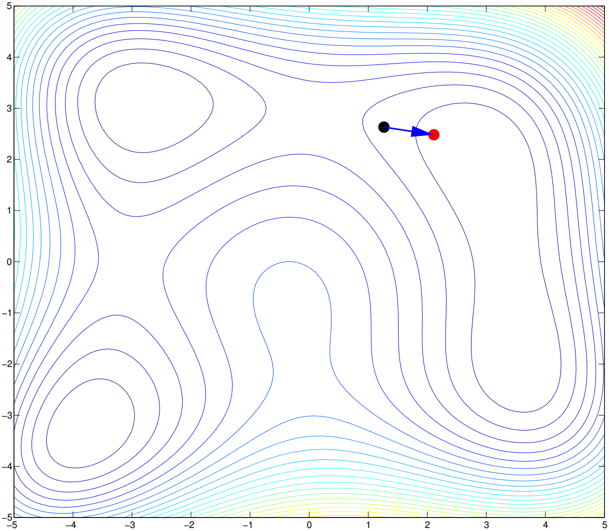
Line Search



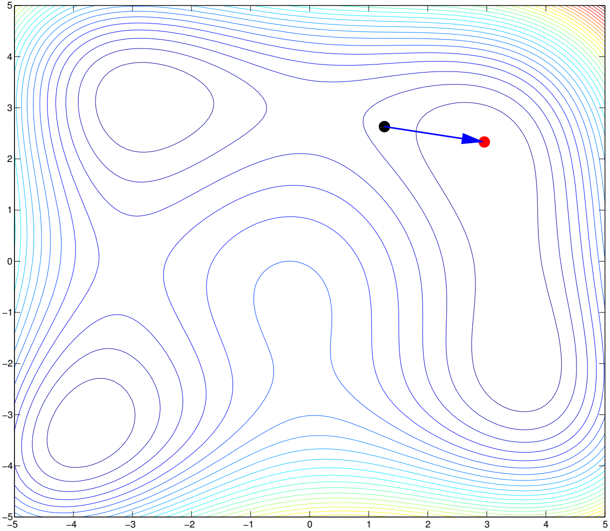
Line Search



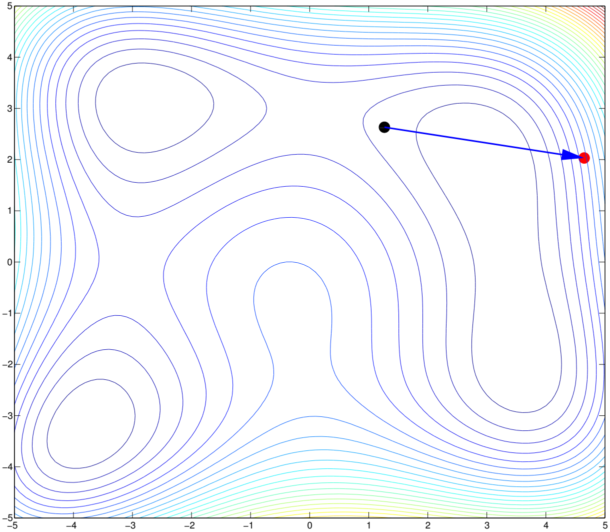
Line Search



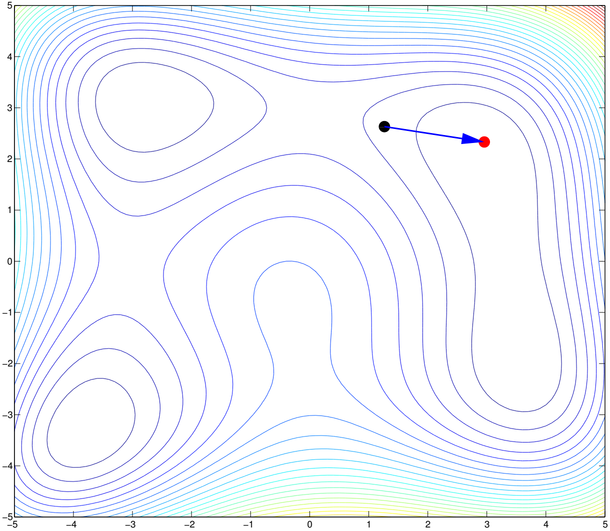
Line Search



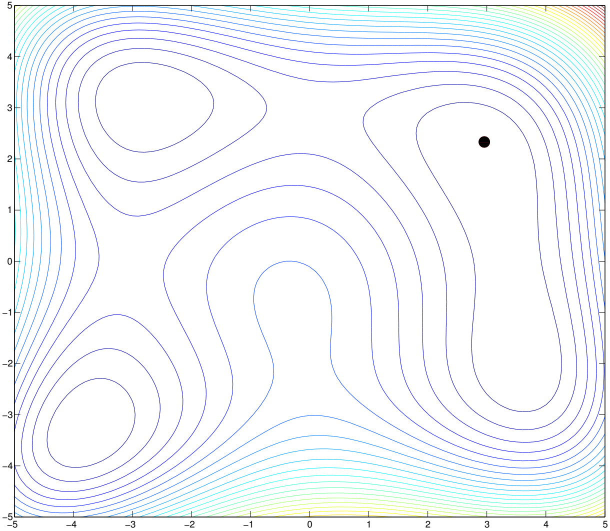
Line Search



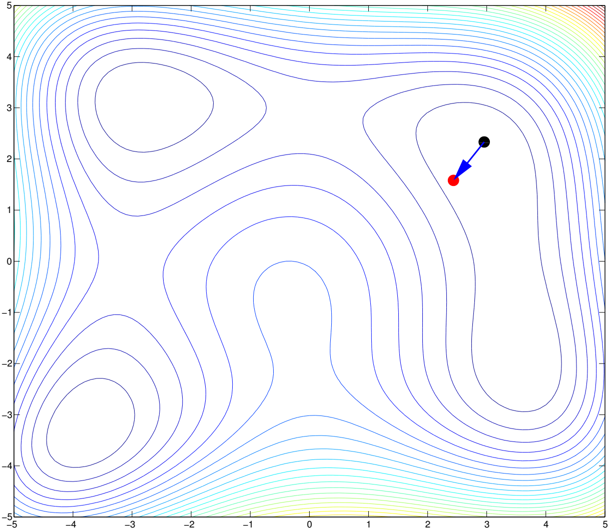
Line Search



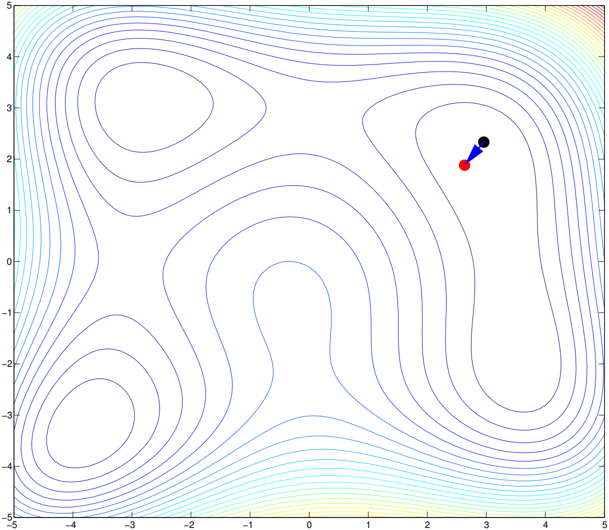
Line Search



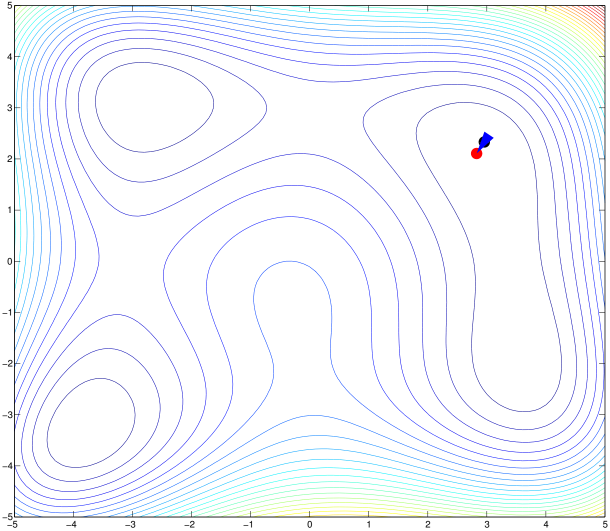
Line Search



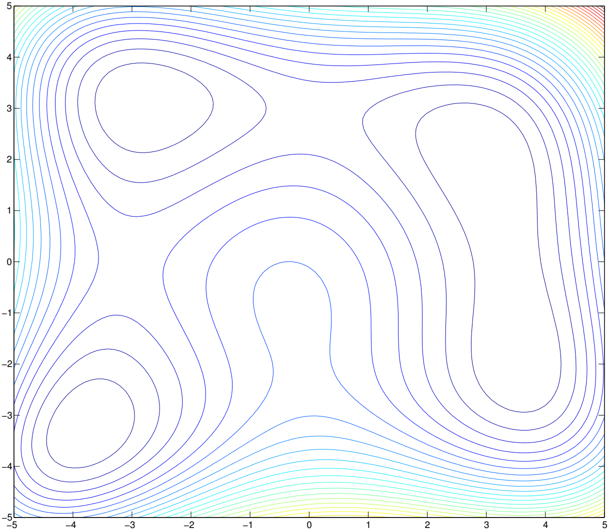
Line Search



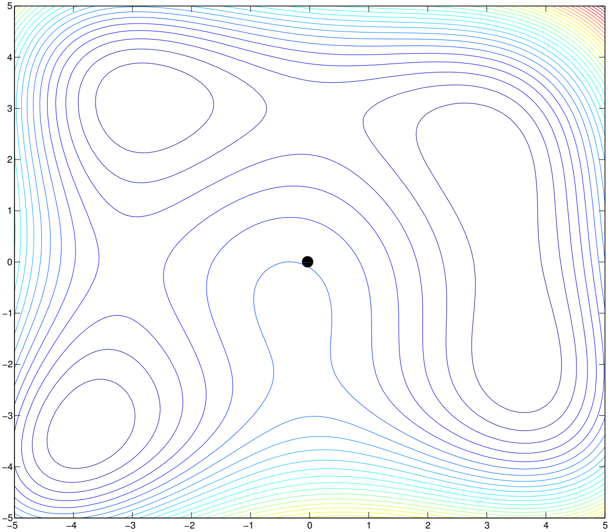
Line Search



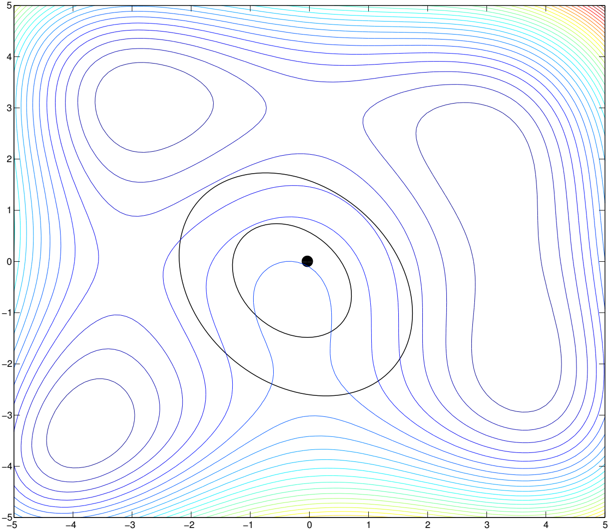
Trust Region Example



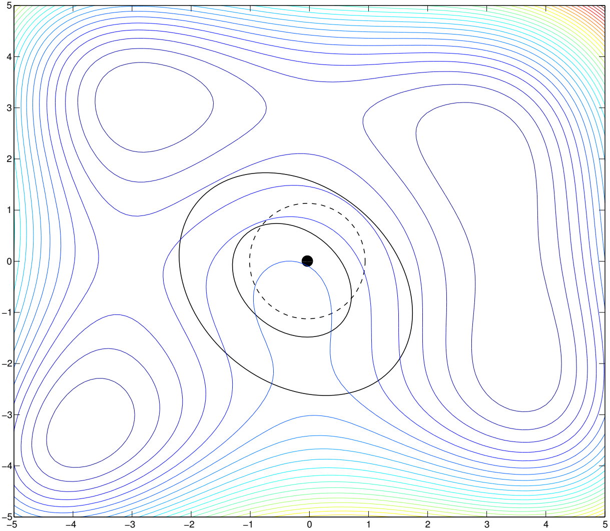
Trust Region Example



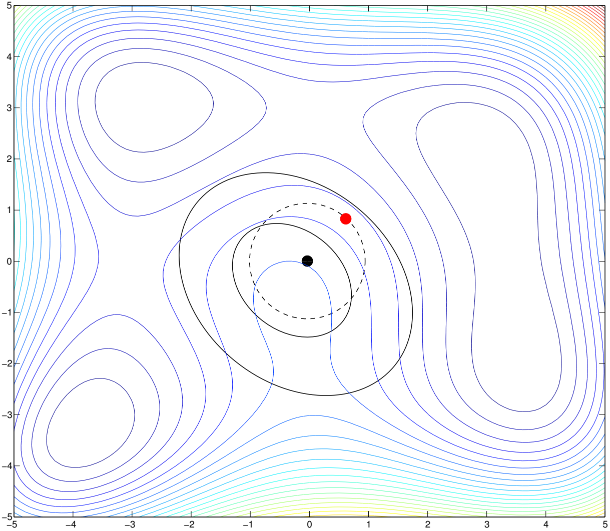
Trust Region Example



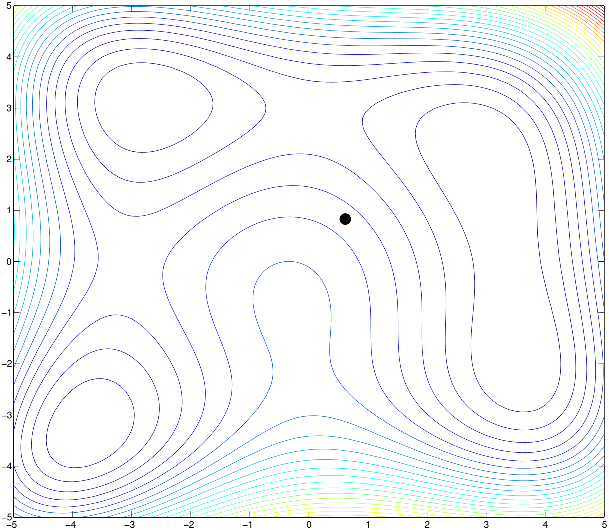
Trust Region Example



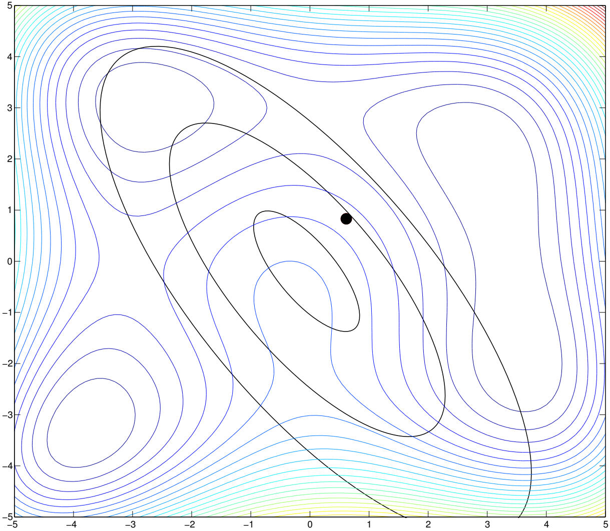
Trust Region Example



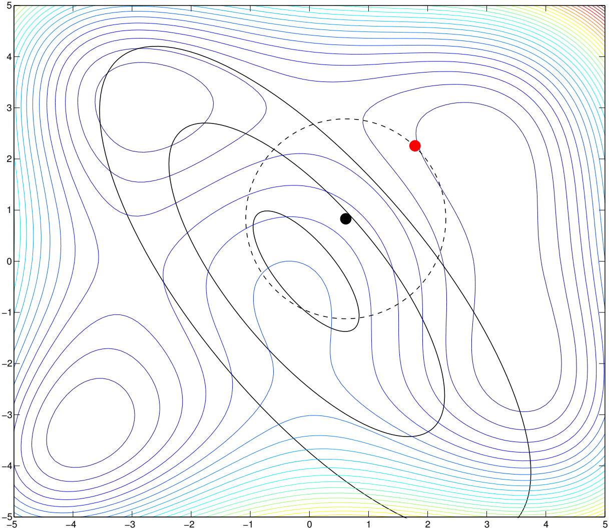
Trust Region Example



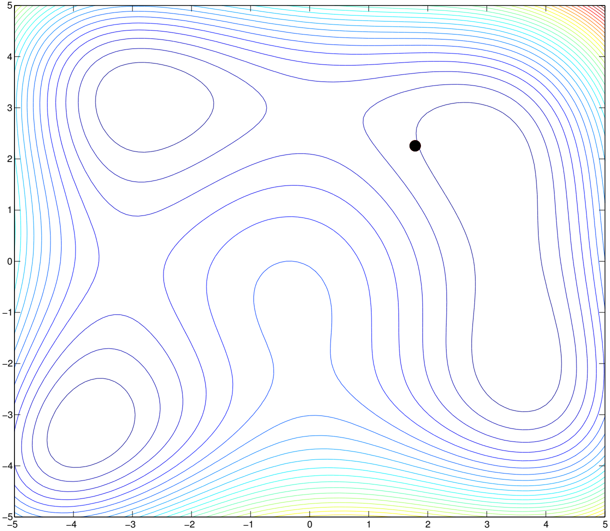
Trust Region Example



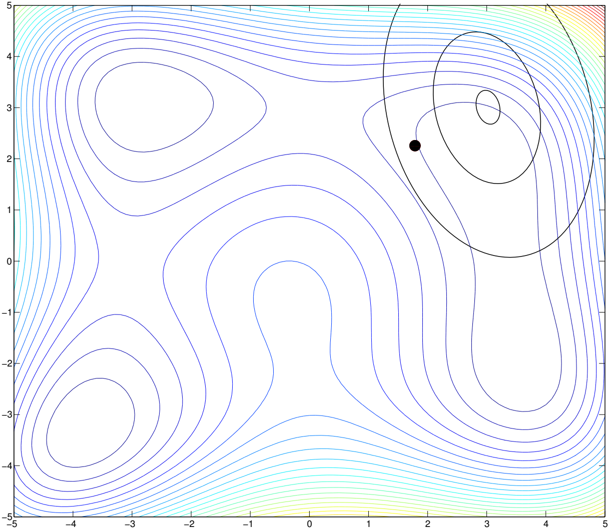
Trust Region Example



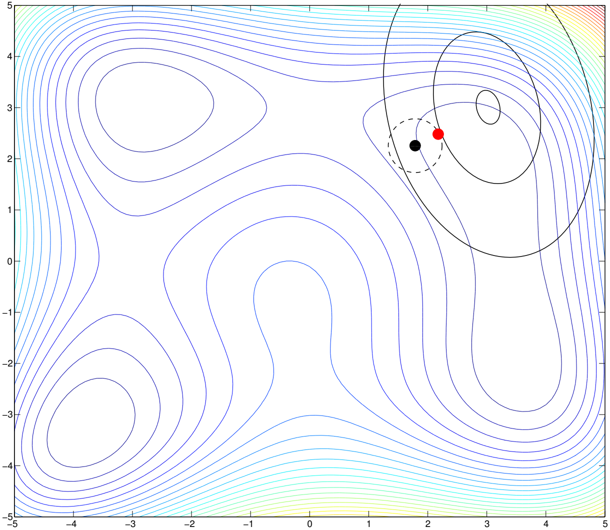
Trust Region Example



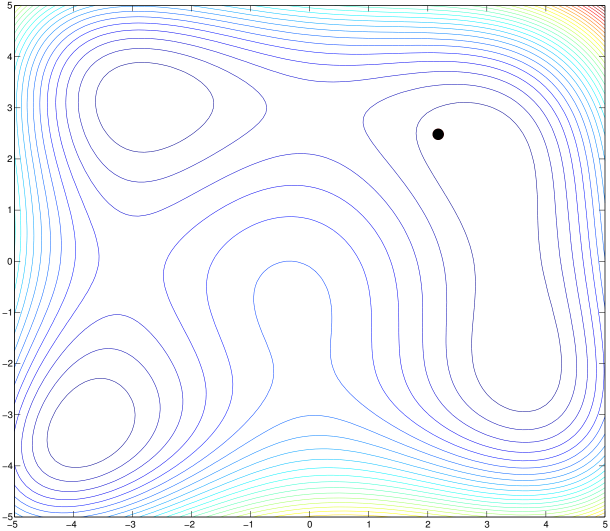
Trust Region Example



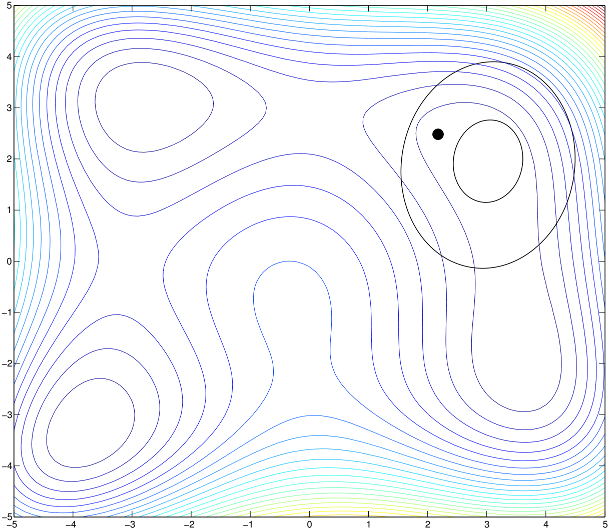
Trust Region Example



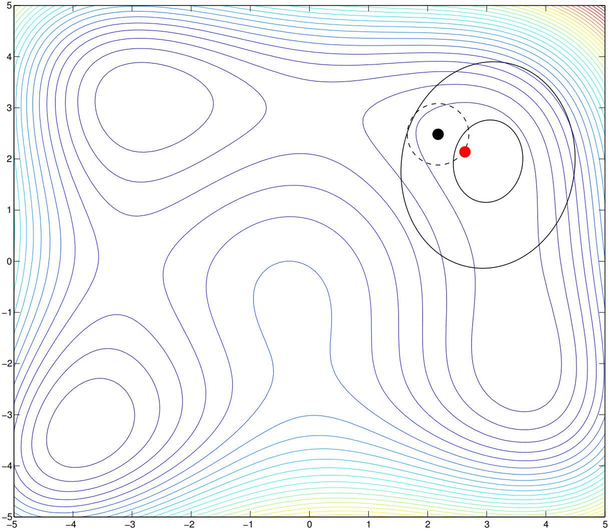
Trust Region Example



Trust Region Example



Trust Region Example



Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \text{ is not available}$$

- These types of problems are called "black-box" optimization problems or derivative-free optimization problems.

Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \text{ is not available}$$

- These types of problems are called “black-box” optimization problems or derivative-free optimization problems.

- Derivatives are traditionally very important in optimization

Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x)$$

$$\text{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \text{ is not available}$$

- These types of problems are called "black-box" optimization problems or derivative-free optimization problems.

- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?

Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \text{ is not available}$$

- These types of problems are called “black-box” optimization problems or derivative-free optimization problems.

- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?
  - What information is contained in the gradient?

Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \text{ is not available}$$

- These types of problems are called "black-box" optimization problems or derivative-free optimization problems.
- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?
  - What information is contained in the gradient?
- As computers become more powerful, more simulations appear

Optimizing simulations

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \textbf{ is not available}$$

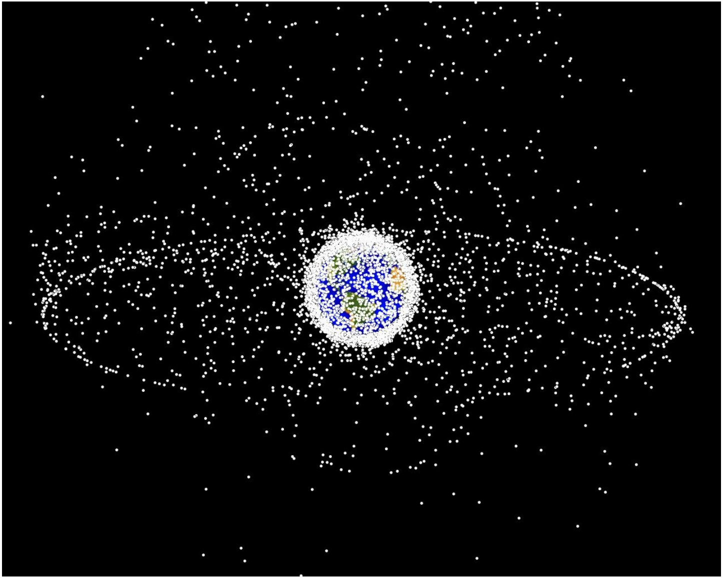
- These types of problems are called "black-box" optimization problems or derivative-free optimization problems.
- Derivatives are traditionally very important in optimization
  - How do you find minima/maxima in calculus?
  - What information is contained in the gradient?
- As computers become more powerful, more simulations appear
- Disclaimer: If the problem has derivatives please use them!

Optimizing simulations

Derivative-free optimization problems show up throughout the world

- Engine design
- Physical experiments
- Tribology
- Satellite planning

Satellite Collision Avoidance



Courtesy NASA/JPL-Caltech

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion
  - You can run a simulation at any duration and time

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion
  - You can run a simulation at any duration and time
  - Each simulation takes ≈ 20 seconds and we have 2.5 days

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion
  - You can run a simulation at any duration and time
  - Each simulation takes ≈ 20 seconds and we have 2.5 days
  - Want a probability of collision under 1 in a billion (10^{-9})

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion
  - You can run a simulation at any duration and time
  - Each simulation takes ≈ 20 seconds and we have 2.5 days
  - Want a probability of collision under 1 in a billion (10^{-9})
  - Use the least amount of fuel as possible

Satellite Collision Avoidance

- Problem setup
  - Receive a warning from an international monitoring agency
  - The probability of collision is $10^{-1.61} \approx 0.02$.
  - Can control the duration and time of your rocket burn
    - Only burn in the direction of motion
  - You can run a simulation at any duration and time
  - Each simulation takes ≈ 20 seconds and we have 2.5 days
  - Want a probability of collision under 1 in a billion (10^{-9})
  - Use the least amount of fuel as possible
- What would you do?

Try it yourself

MATLAB

Try it yourself

MATLAB

Process should hopefully be described to a computer.

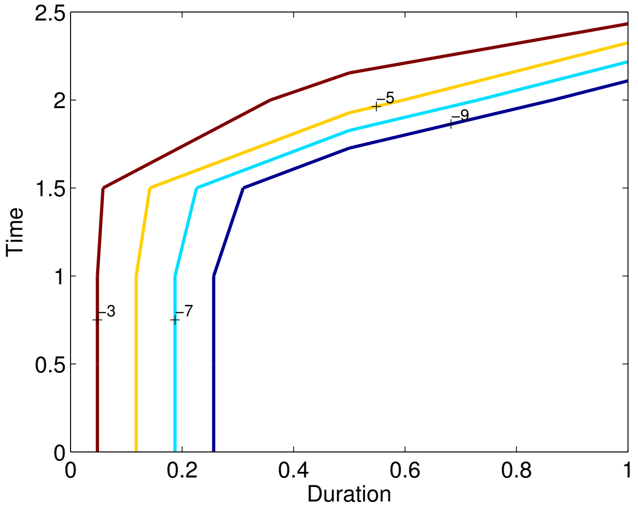
Common first thoughts

- Random Guessing
  - Not very elegant
  - Evaluate points close to previous points
  - Needs some sort of finishing step

Common first thoughts

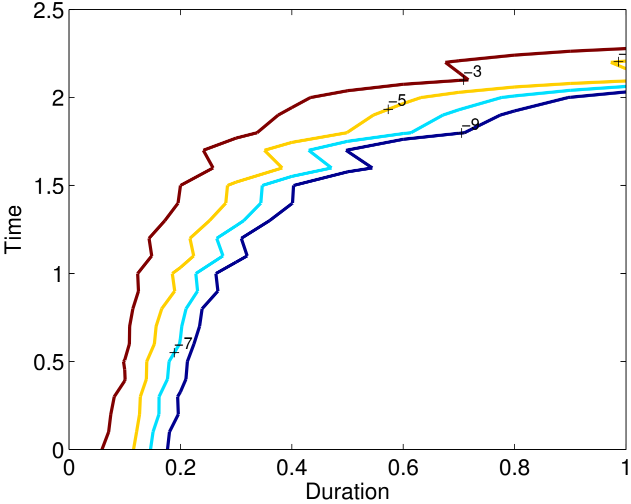
- Random Guessing
  - Not very elegant
  - Evaluate points close to previous points
  - Needs some sort of finishing step
- Finite-differences can be problematic
  - $n + 1$ or $2n + 1$ evaluations at every point
  - Simulations are often noisy

Grid Search



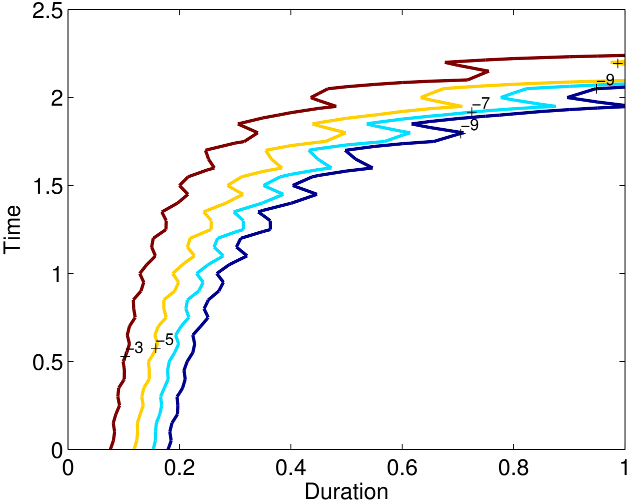
[0:0.5:1] by [0:0.5:2.5] $\approx$ 6 minutes

Grid Search



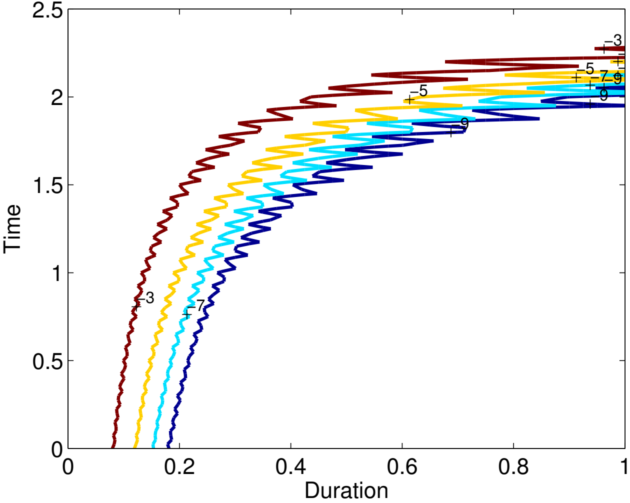
[0:0.1:1] by [0:0.1:2.5] $\approx$ 1.5 hours

Grid Search



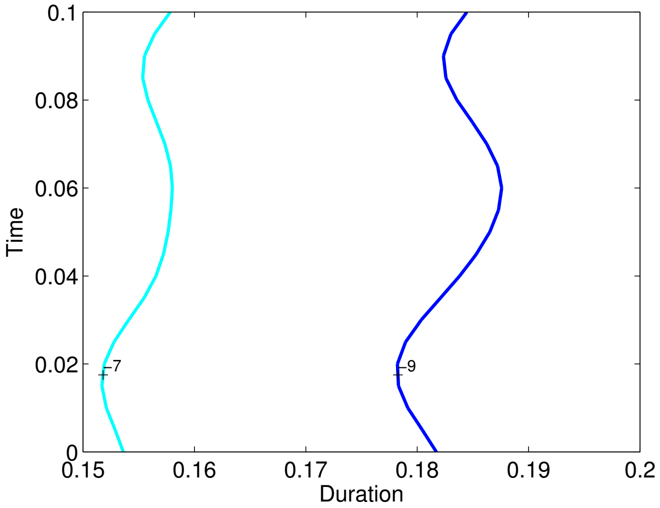
[0:0.05:1] by [0:0.05:2.5] $\approx$ 6 hours

Grid Search



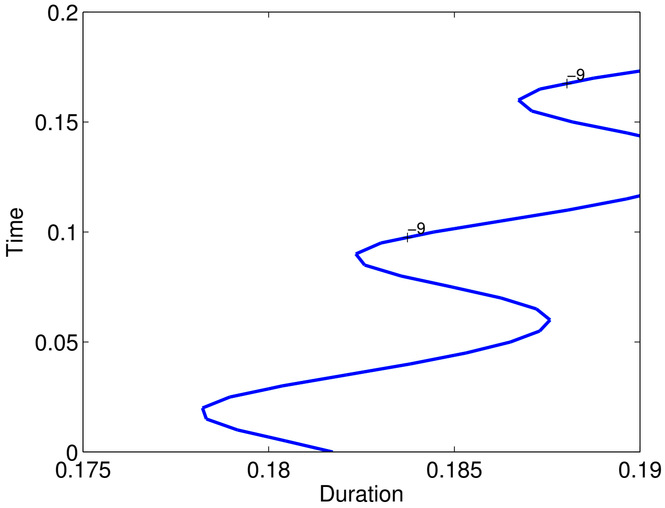
[0:0.025:1] by [0:0.025:2.5] $\approx$ 1 day

Grid Search



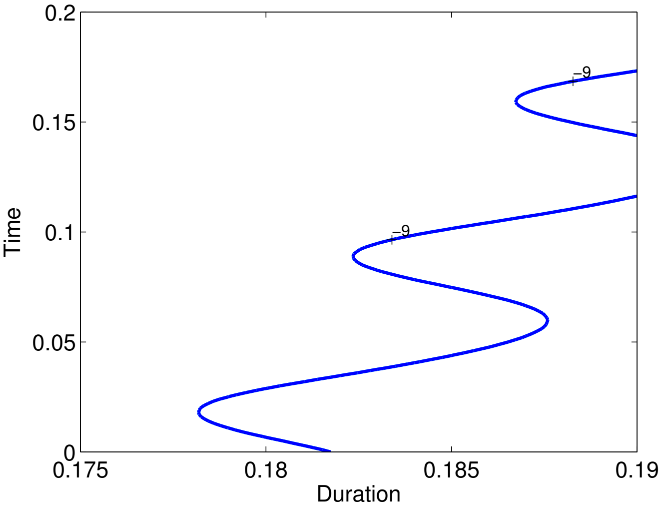
[0.15:0.005:0.2] by [0:0.005:0.1] $\approx$ 1.3 hours

Grid Search



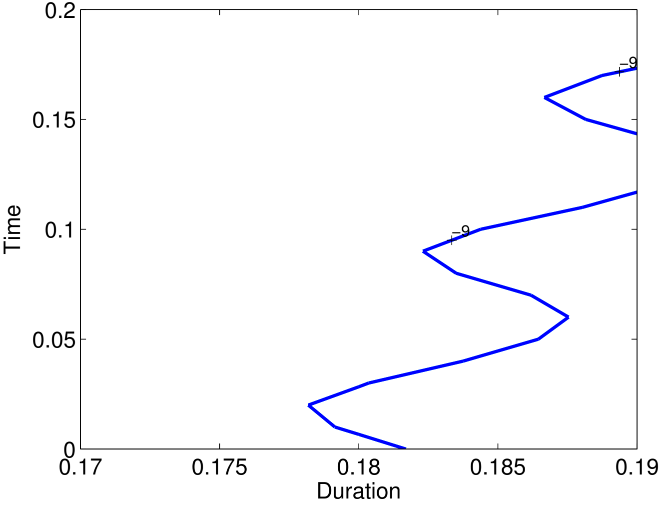
[0.175:0.005:0.19] by [0:0.005:0.2] $\approx$ 55 minutes

Grid Search



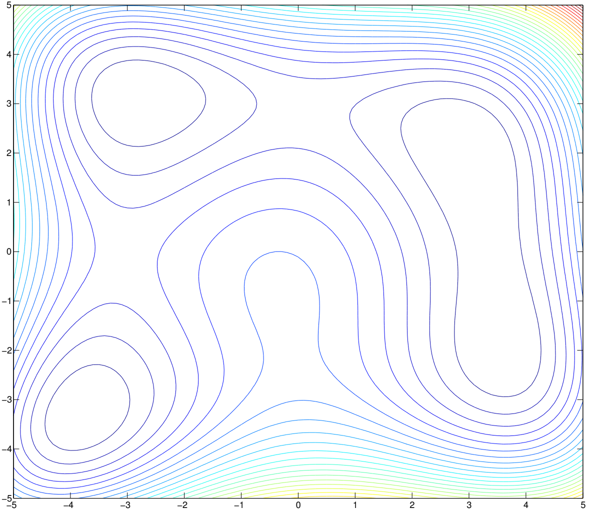
[0.175:0.001:0.19] by [0:0.001:0.2] $\approx$ 18 hours

Grid Search

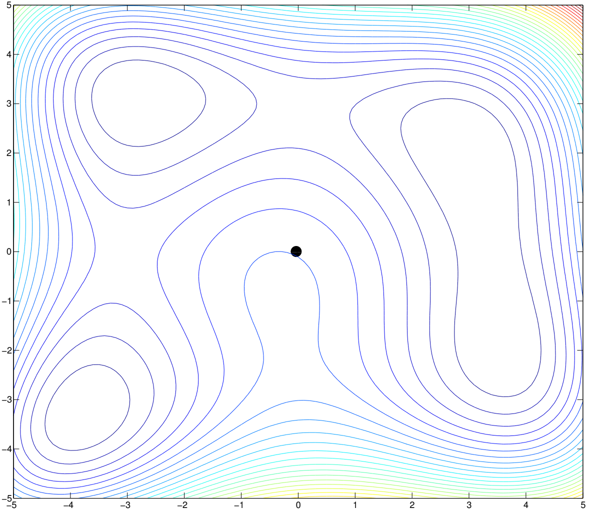


[0.17:0.01:0.19] by [0:0.01:0.2] $\approx$ 20 minutes

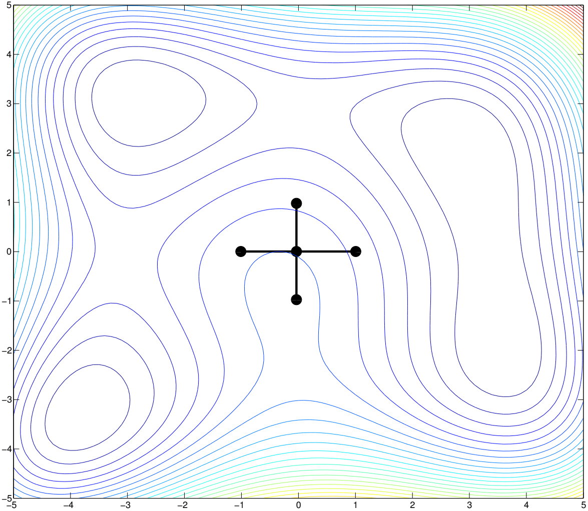
Coordinate Search



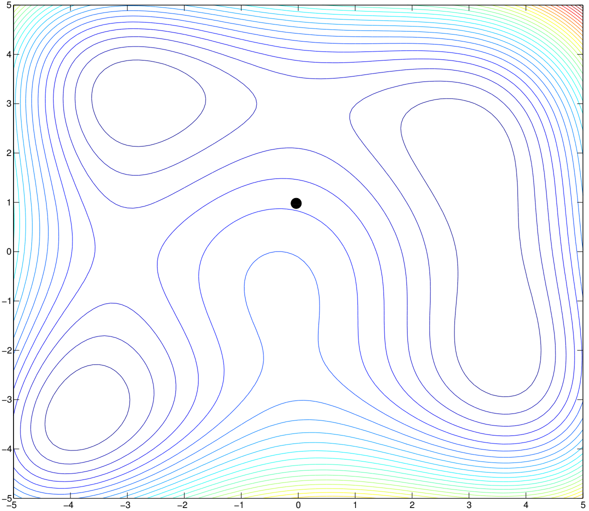
Coordinate Search



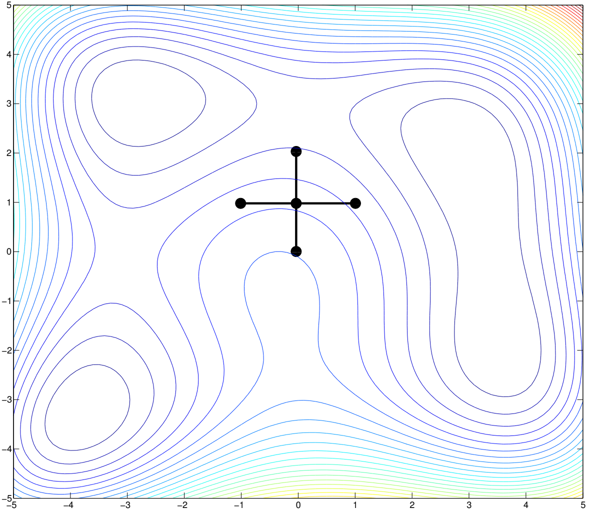
Coordinate Search



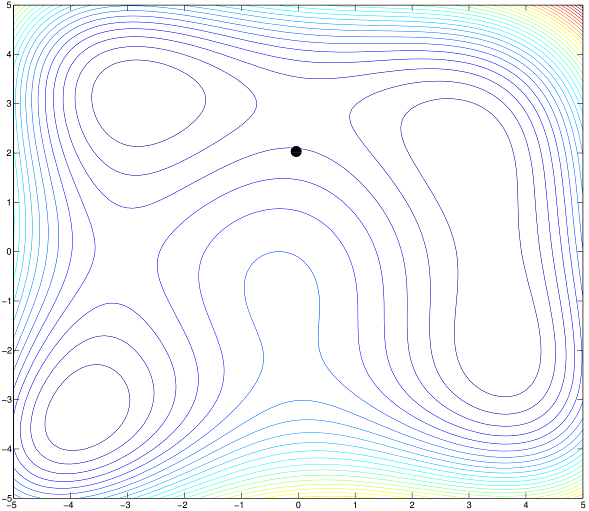
Coordinate Search



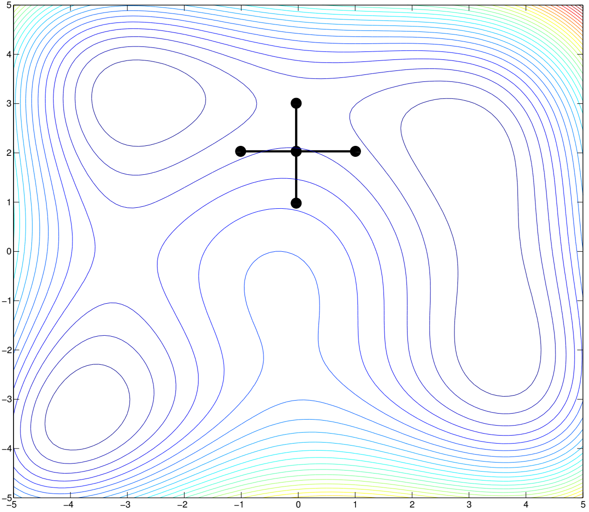
Coordinate Search



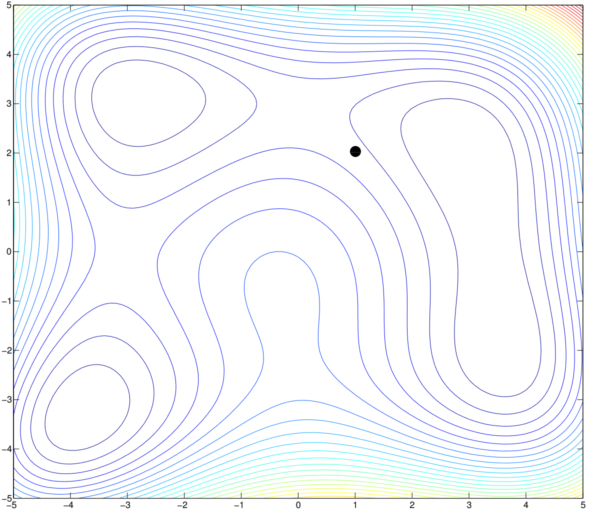
Coordinate Search



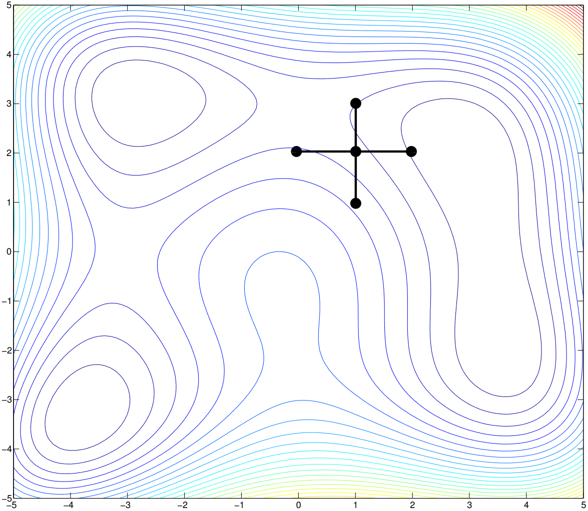
Coordinate Search



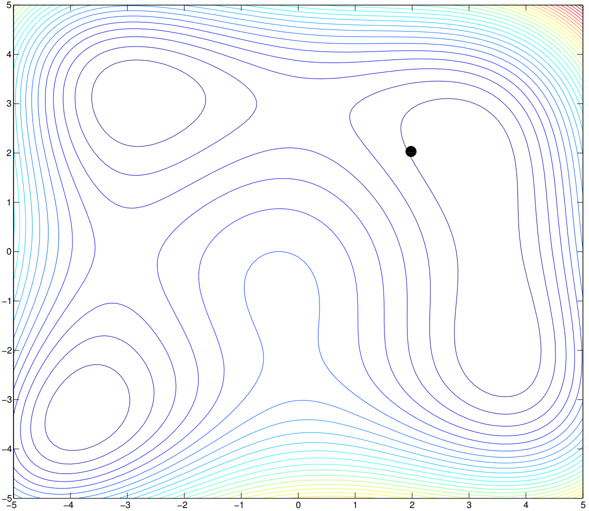
Coordinate Search



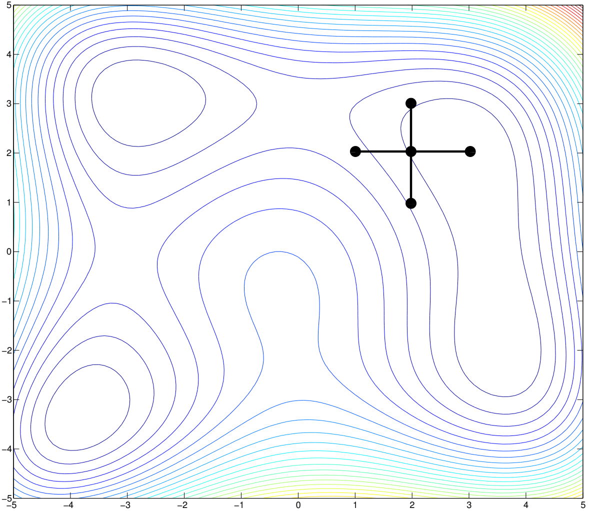
Coordinate Search



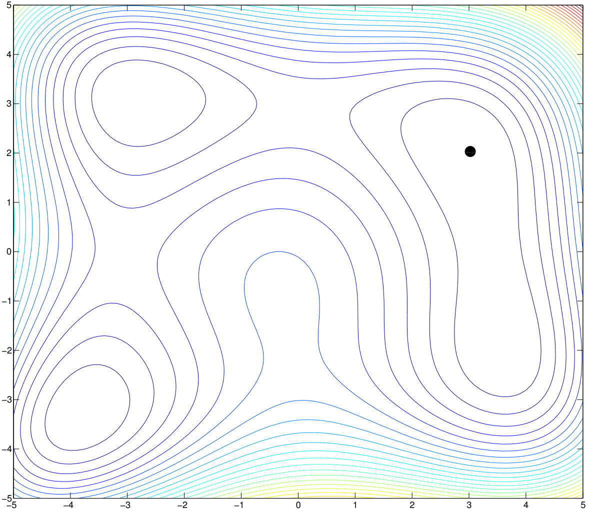
Coordinate Search



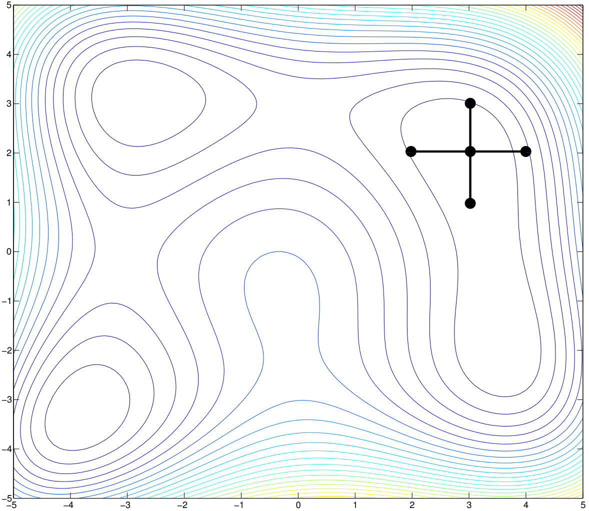
Coordinate Search



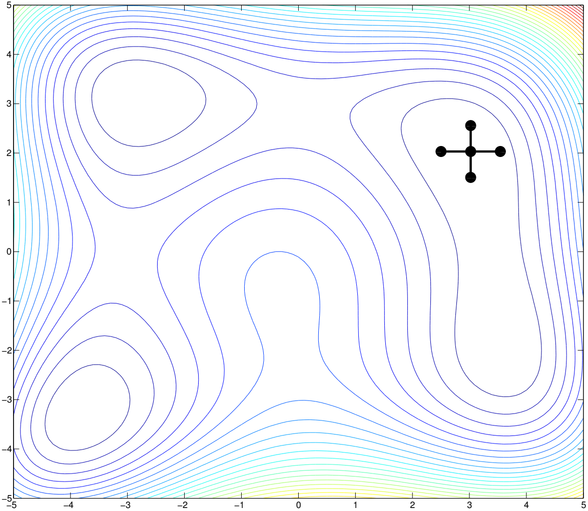
Coordinate Search



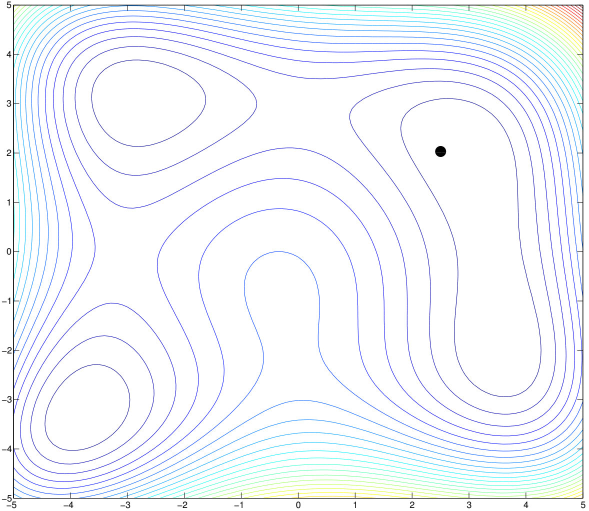
Coordinate Search



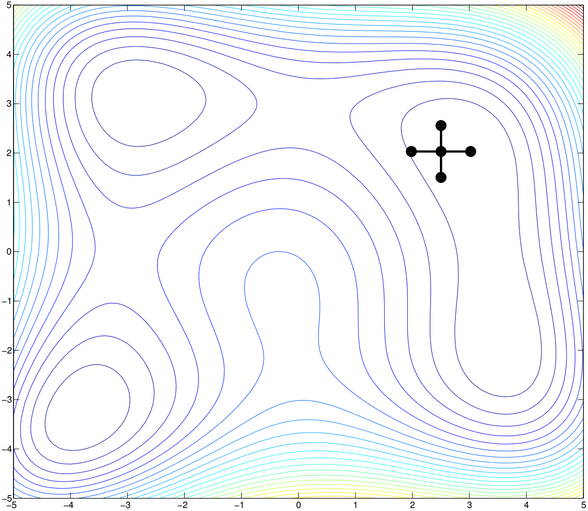
Coordinate Search



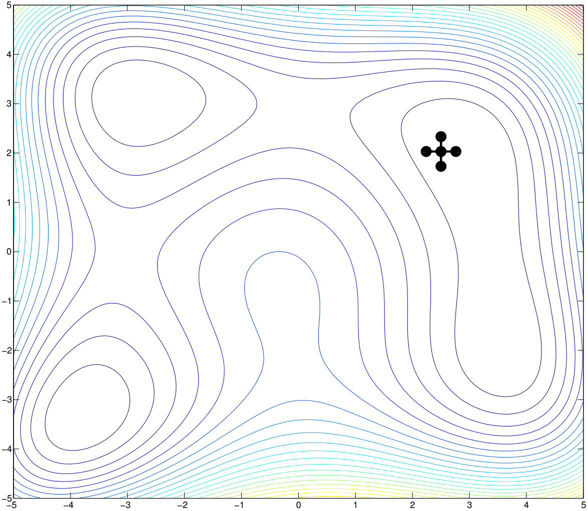
Coordinate Search



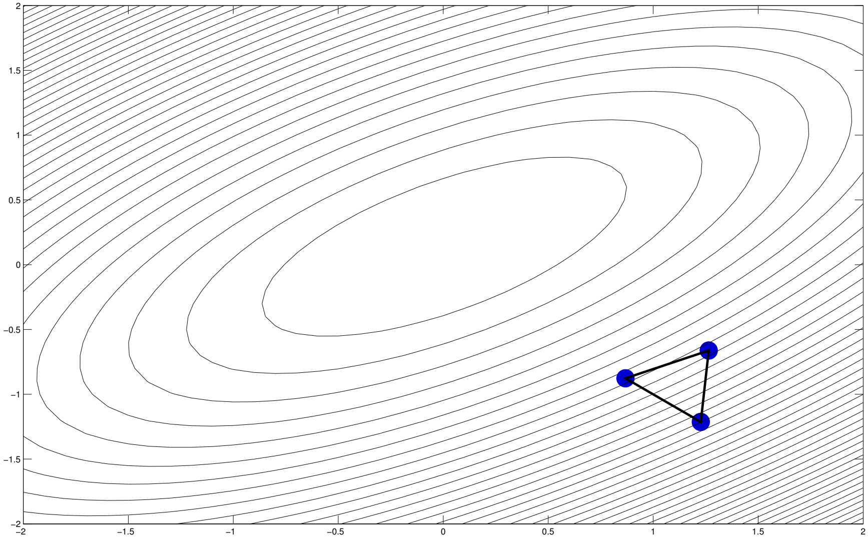
Coordinate Search



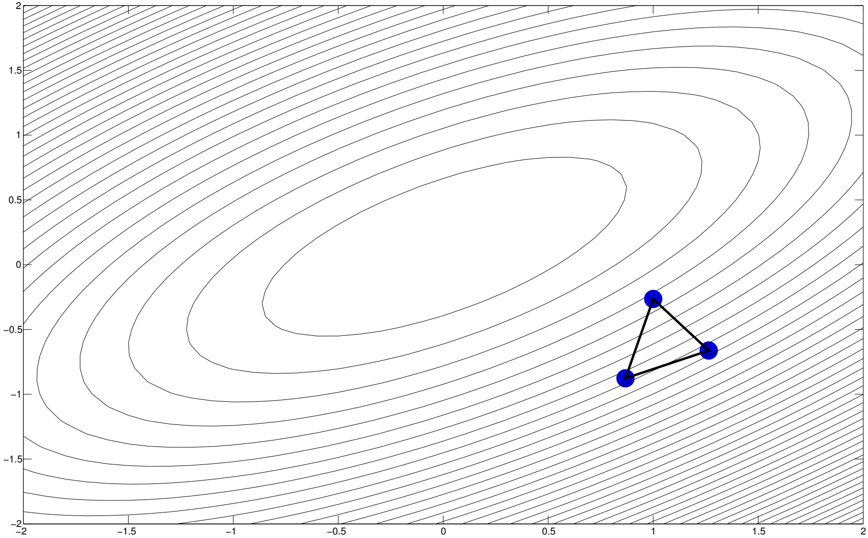
Coordinate Search



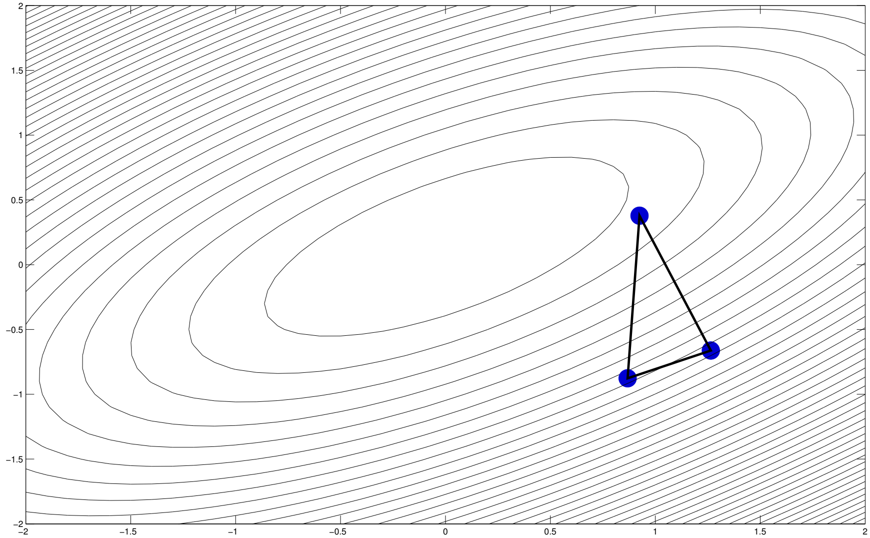
Nelder-Mead



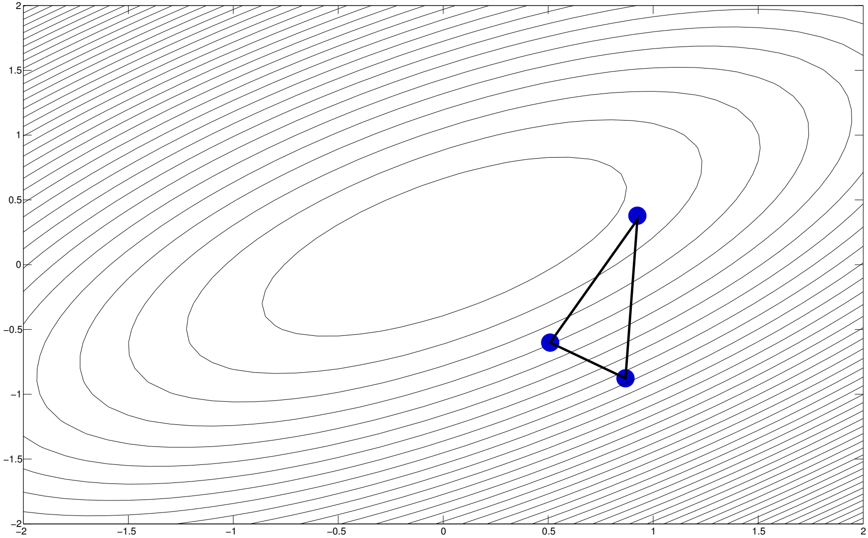
Nelder-Mead



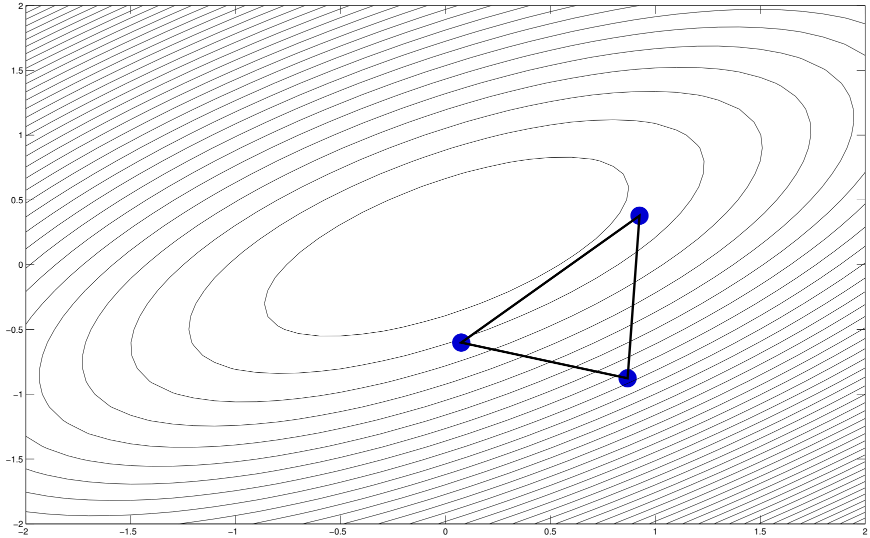
Nelder-Mead



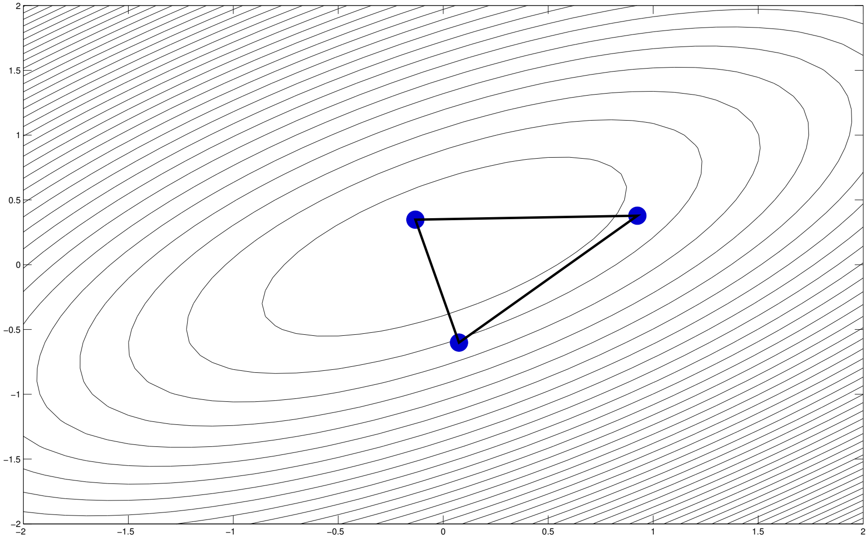
Nelder-Mead



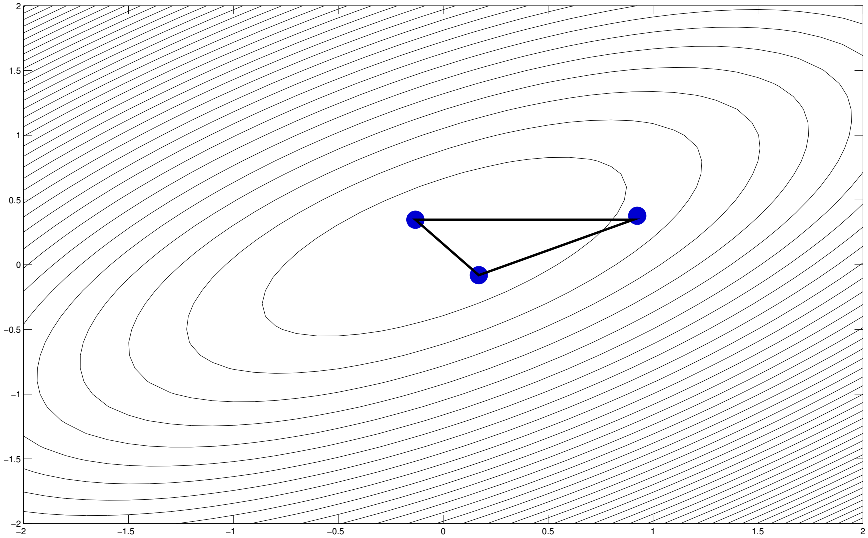
Nelder-Mead



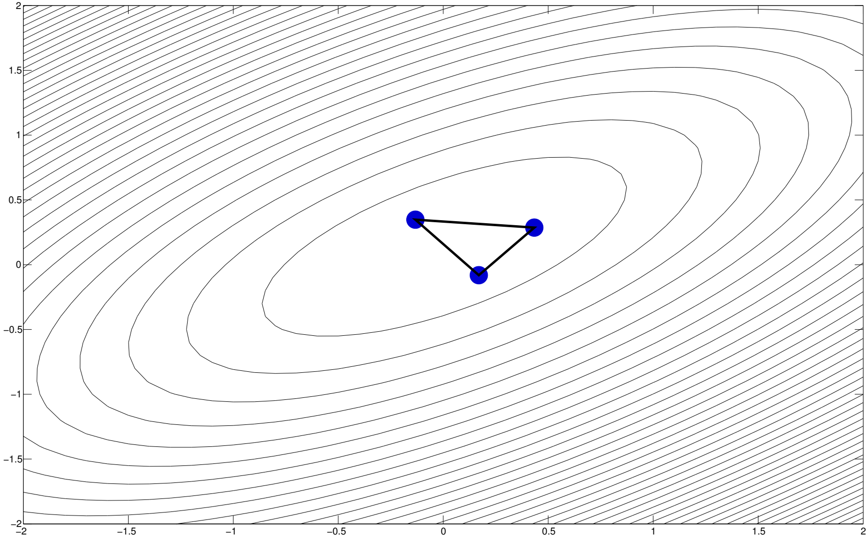
Nelder-Mead



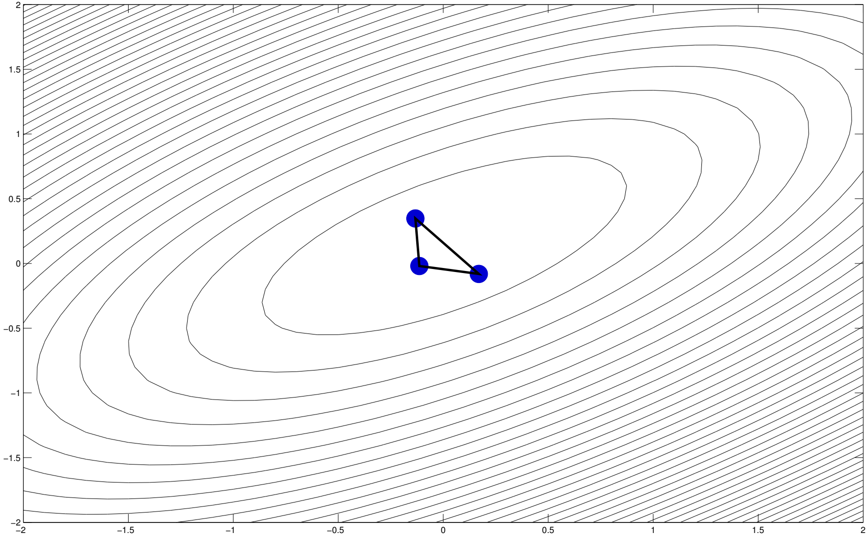
Nelder-Mead



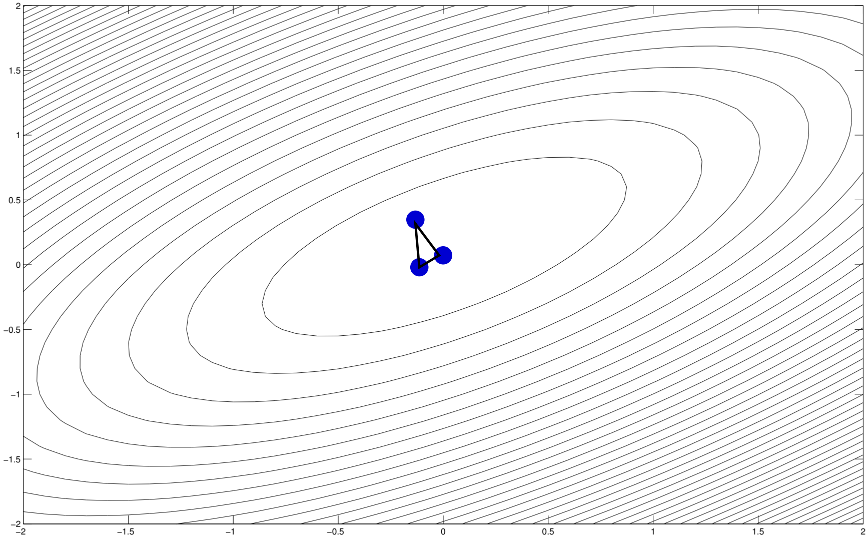
Nelder-Mead



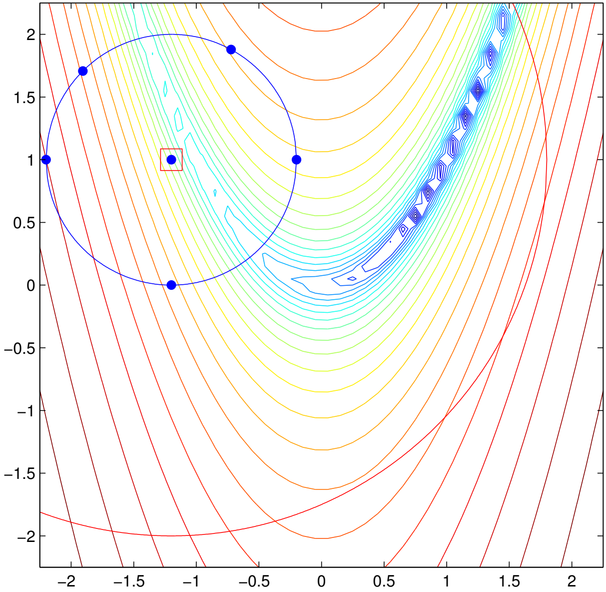
Nelder-Mead



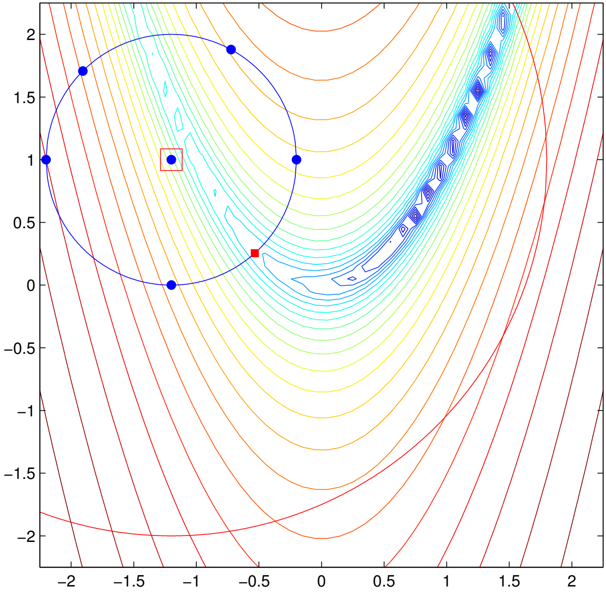
Nelder-Mead



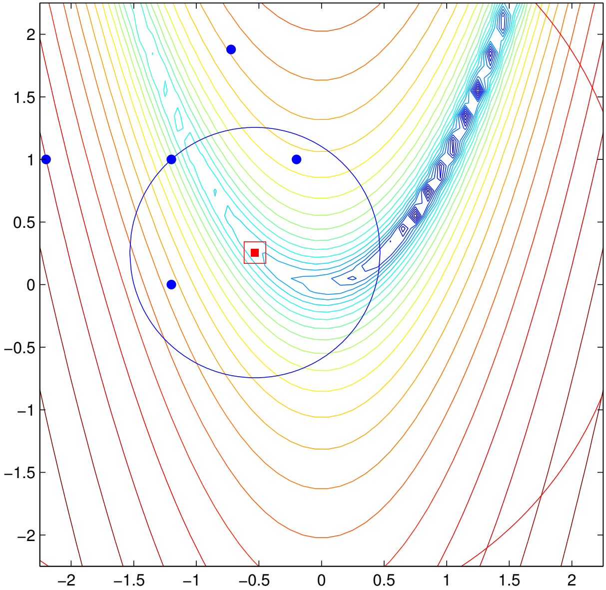
Model-based methods - Interpolation



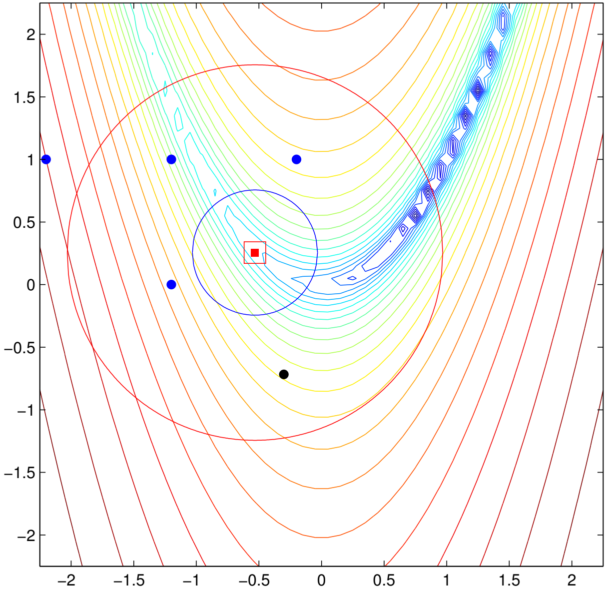
Model-based methods - Interpolation



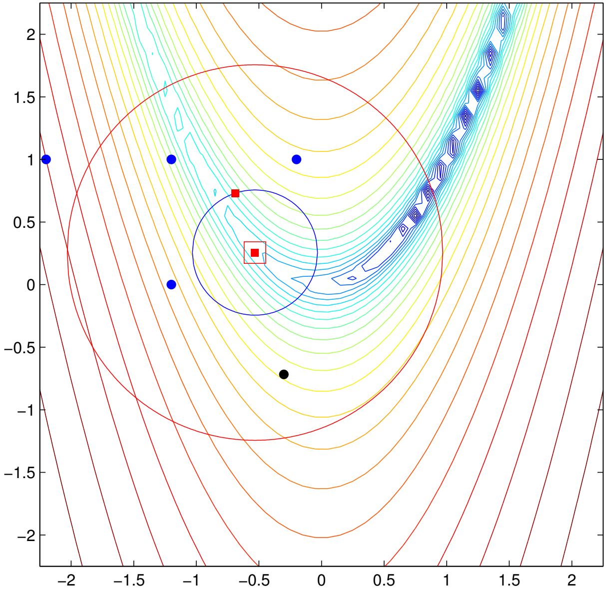
Model-based methods - Interpolation



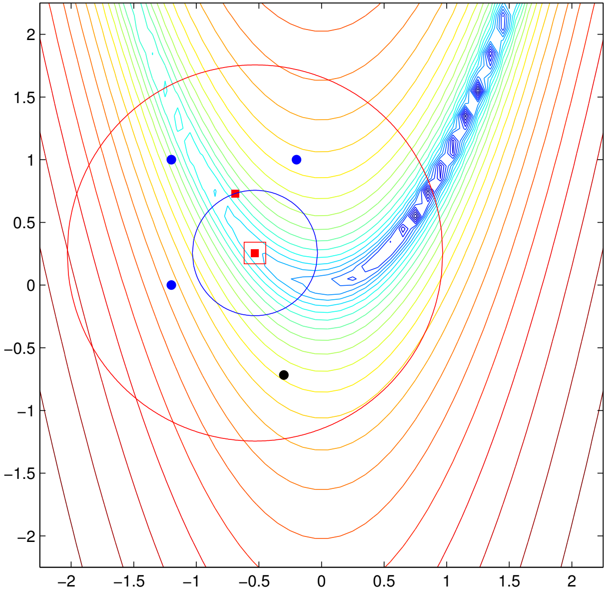
Model-based methods - Interpolation



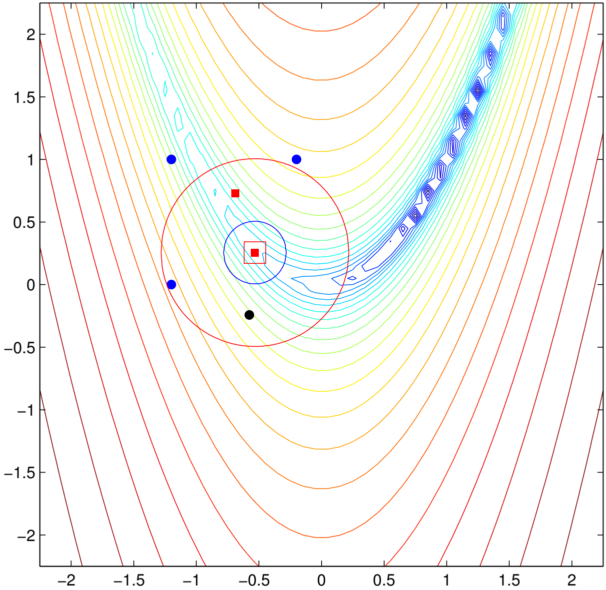
Model-based methods - Interpolation



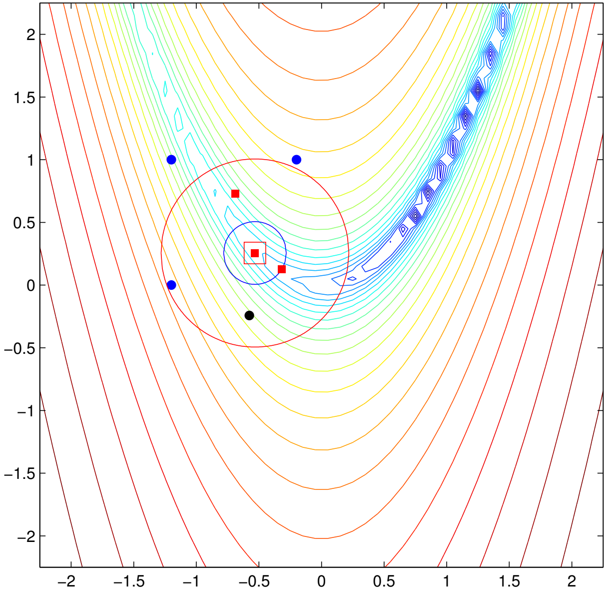
Model-based methods - Interpolation



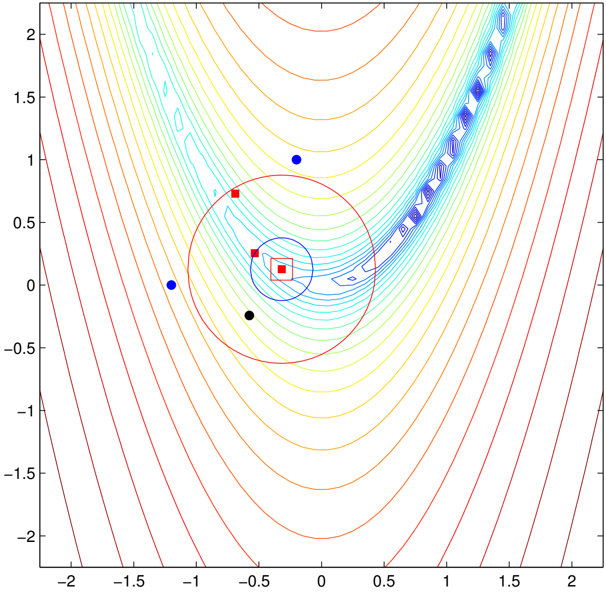
Model-based methods - Interpolation



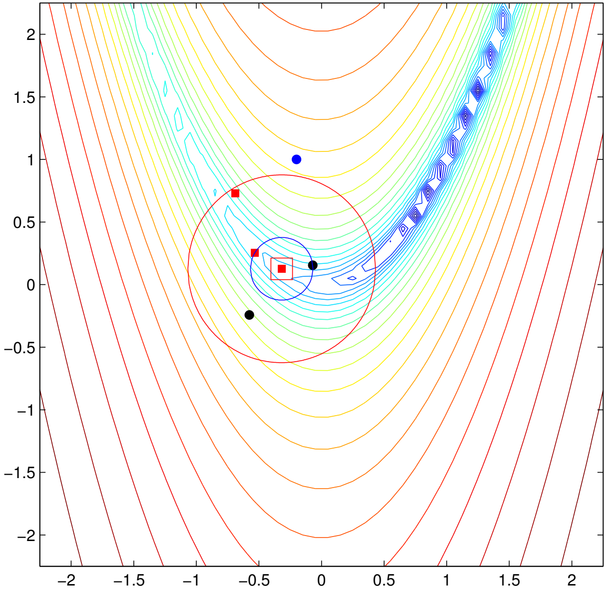
Model-based methods - Interpolation



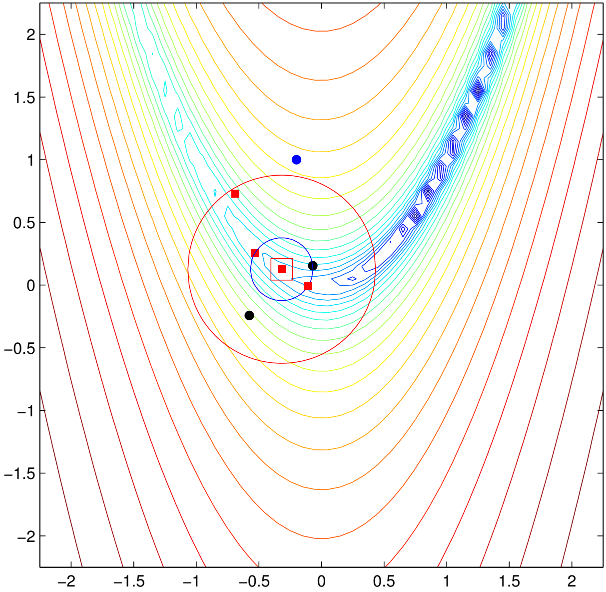
Model-based methods - Interpolation



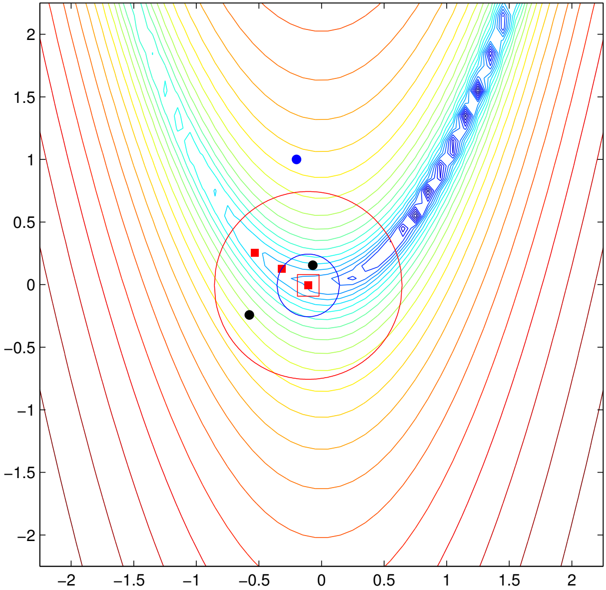
Model-based methods - Interpolation



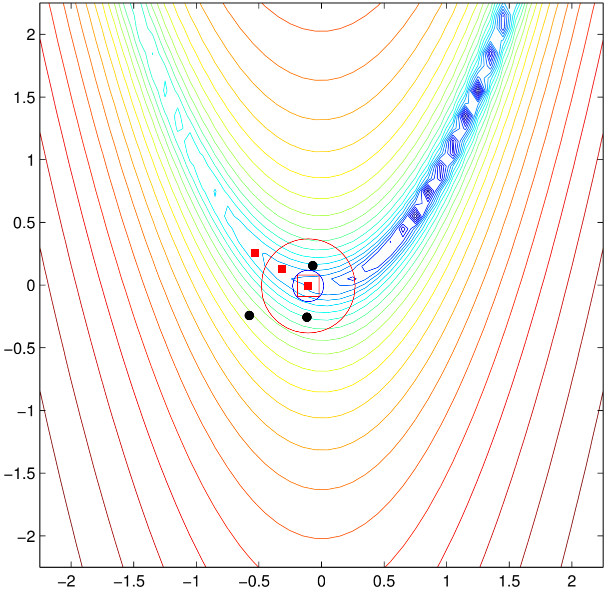
Model-based methods - Interpolation



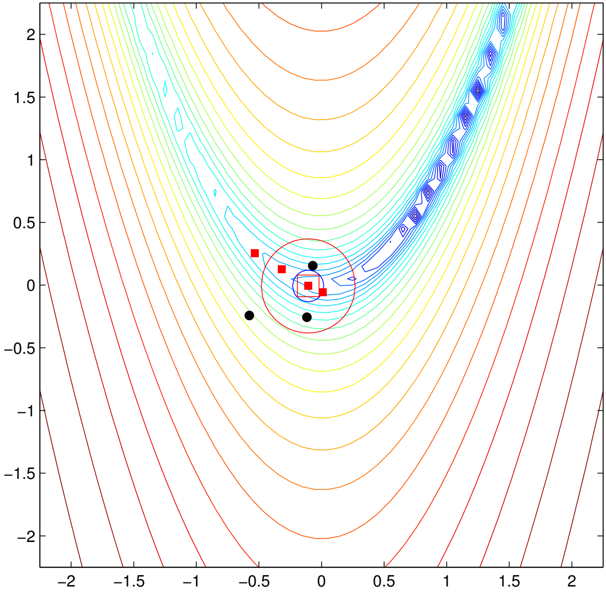
Model-based methods - Interpolation



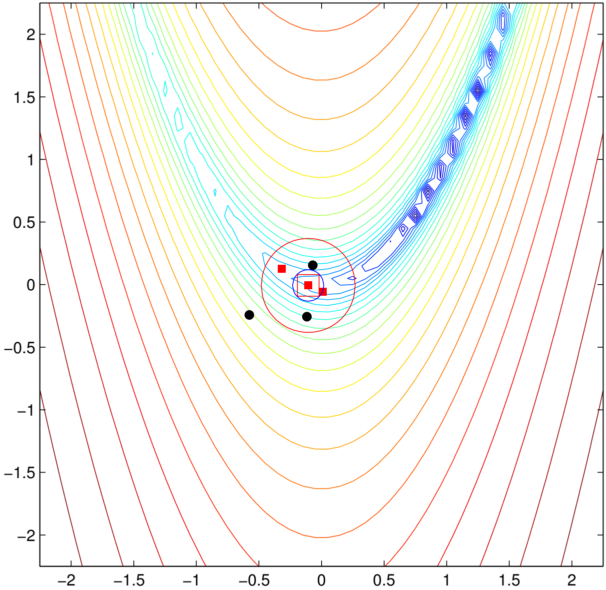
Model-based methods - Interpolation



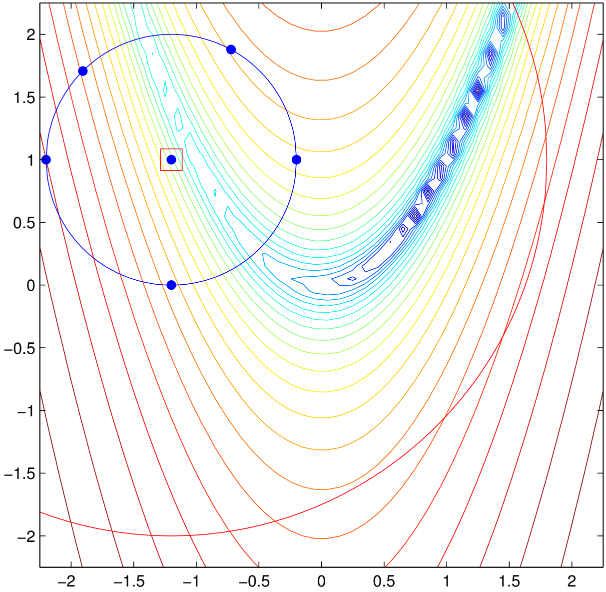
Model-based methods - Interpolation



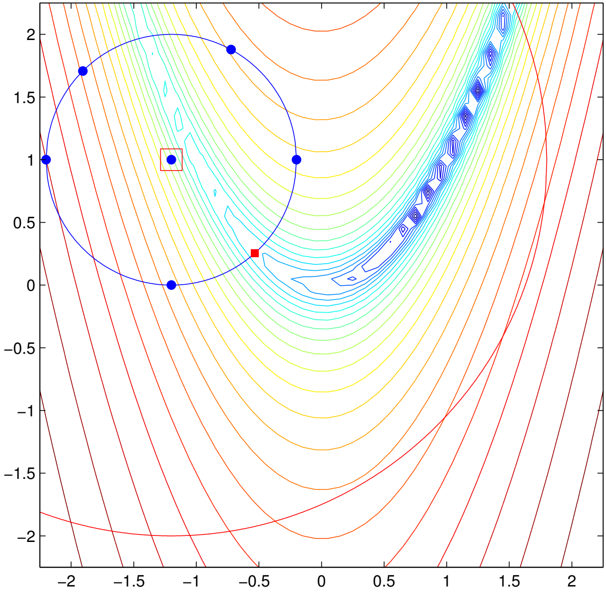
Model-based methods - Interpolation



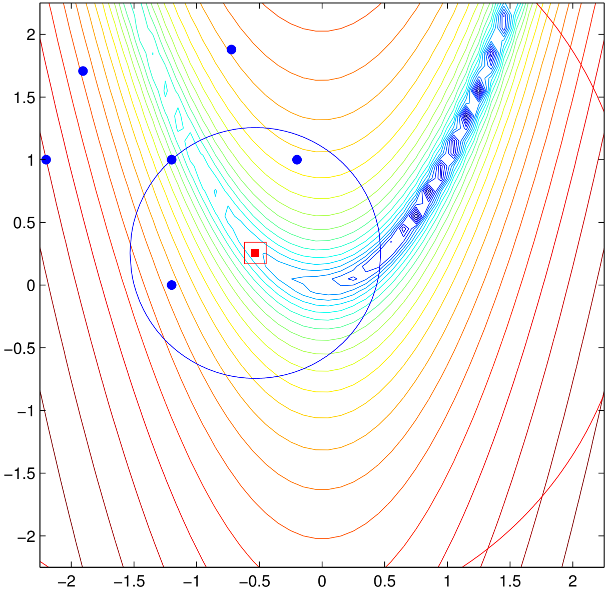
Model-based methods - Regression



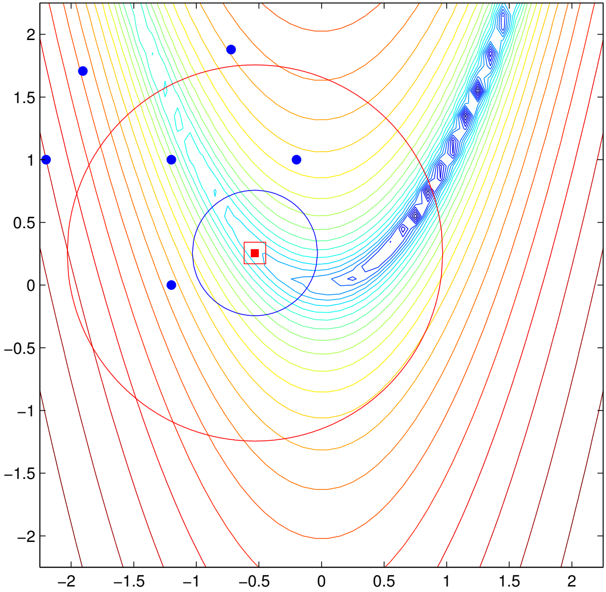
Model-based methods - Regression



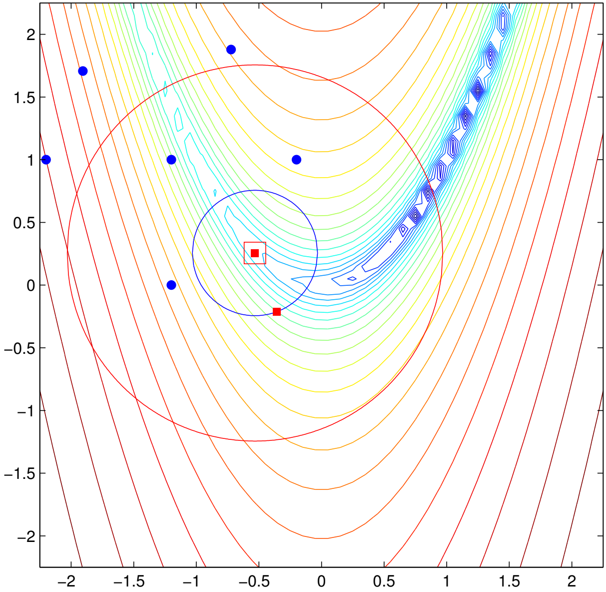
Model-based methods - Regression



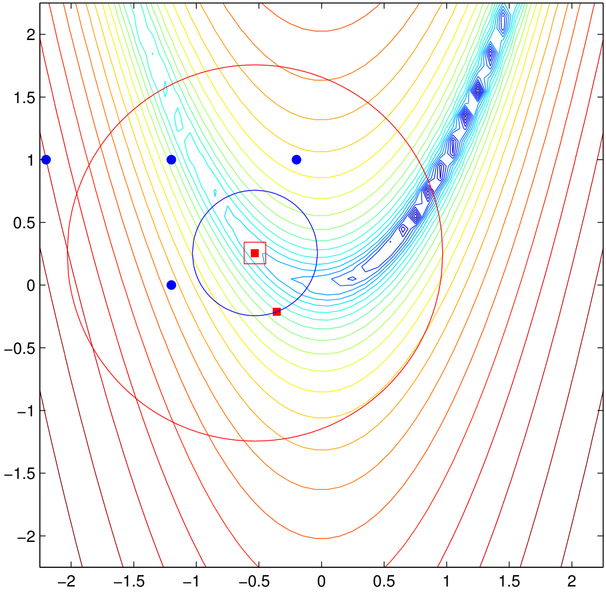
Model-based methods - Regression



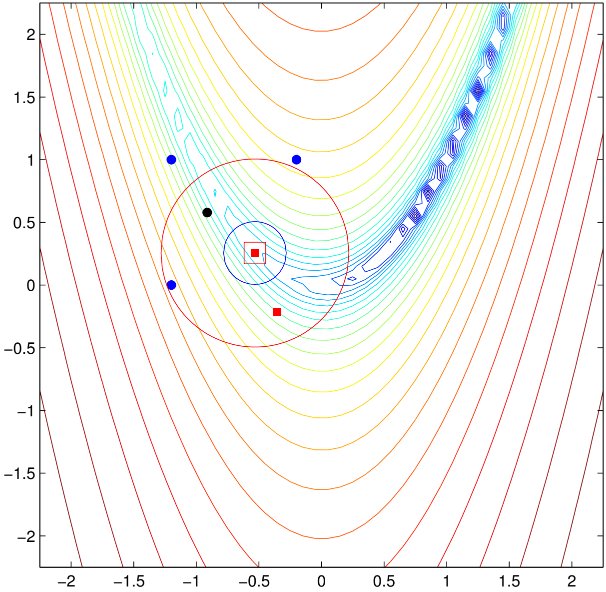
Model-based methods - Regression



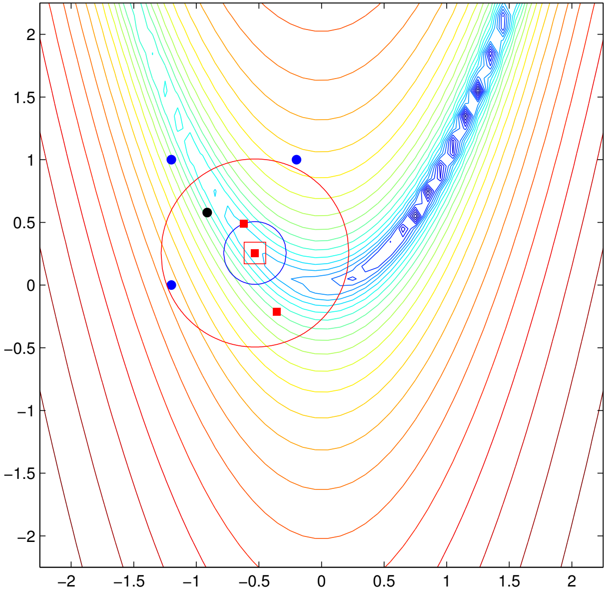
Model-based methods - Regression



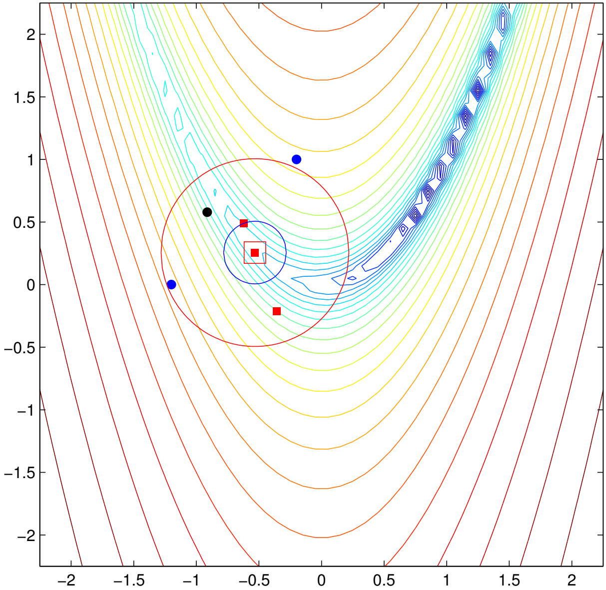
Model-based methods - Regression



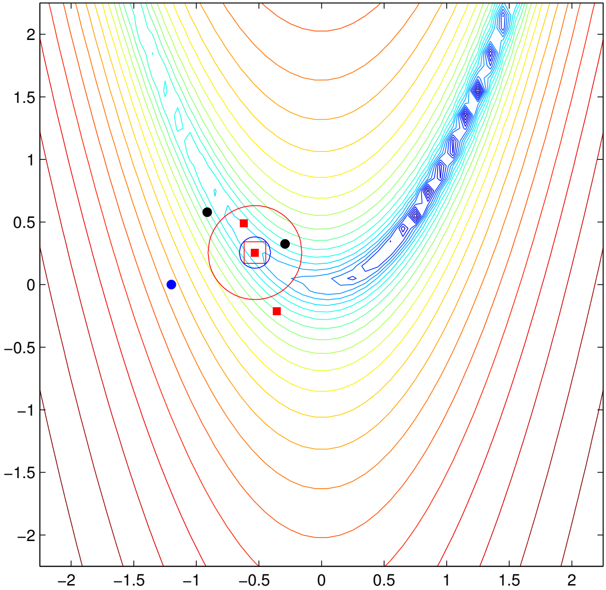
Model-based methods - Regression



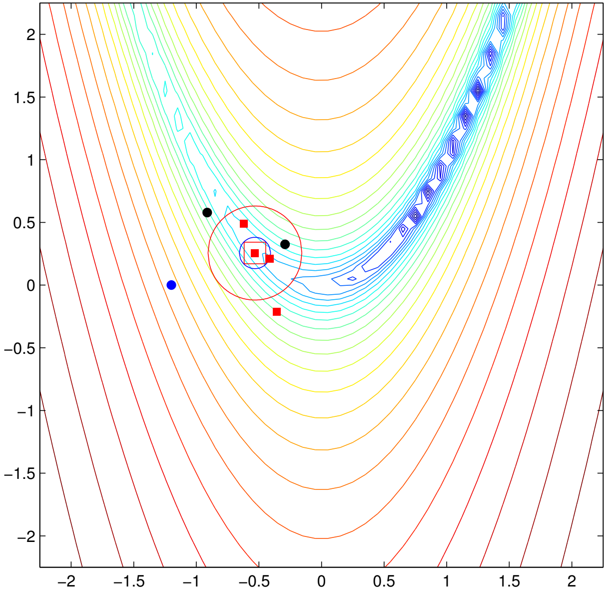
Model-based methods - Regression



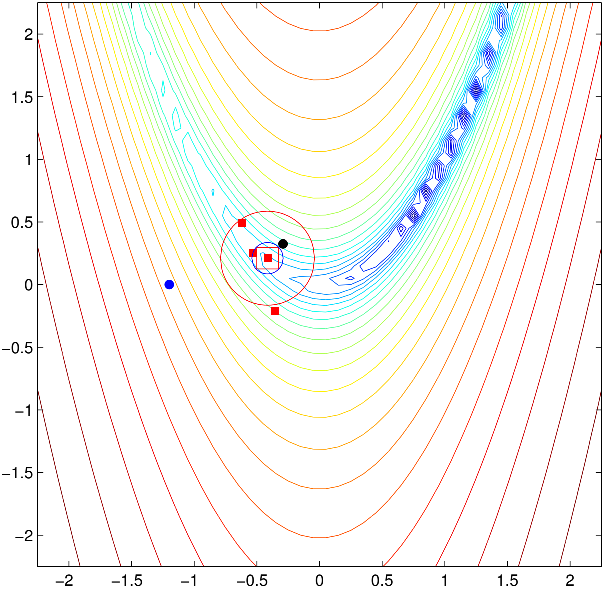
Model-based methods - Regression



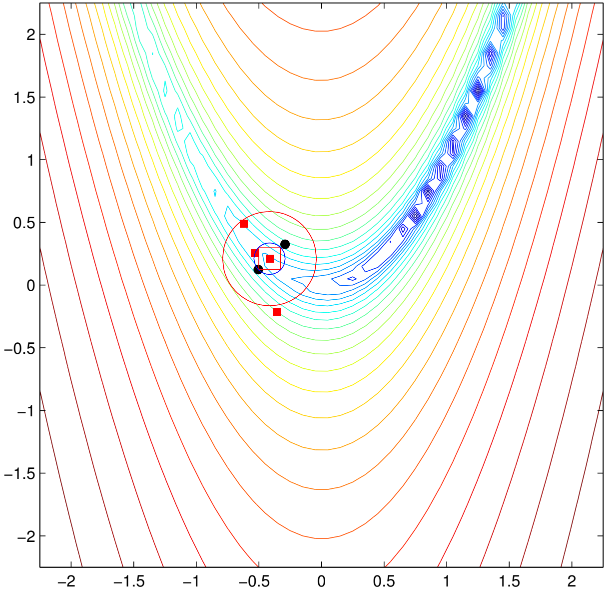
Model-based methods - Regression



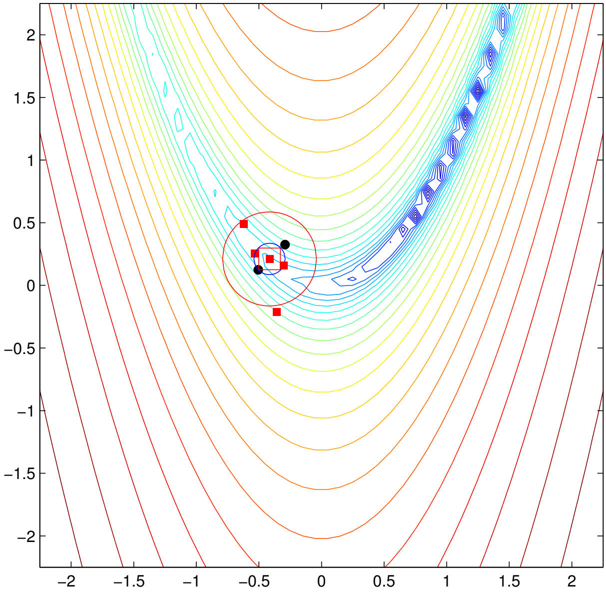
Model-based methods - Regression



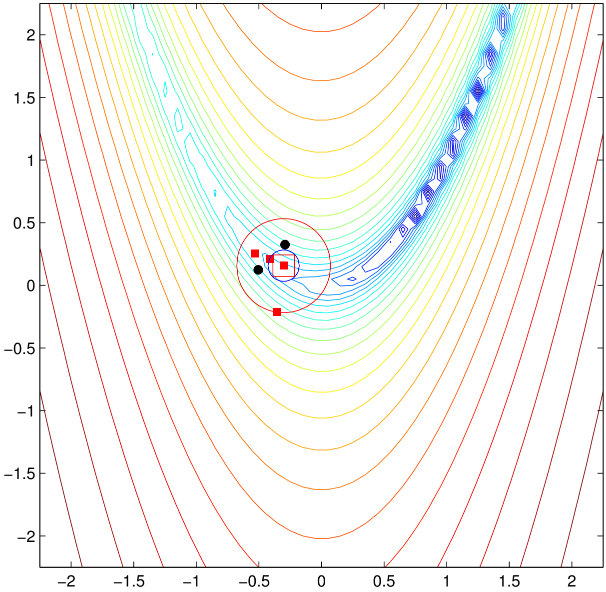
Model-based methods - Regression



Model-based methods - Regression



Model-based methods - Regression



Other options

- Build a model of the entire space (Kriging)

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search

Other options

- Build a model of the entire space (Kriging)

- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search
  - Bacterial Colony Optimization

Other options

- Build a model of the entire space (Kriging)
- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search
  - Bacterial Colony Optimization
  - Grey Wolf Optimization

Other options

- Build a model of the entire space (Kriging)

- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search
  - Bacterial Colony Optimization
  - Grey Wolf Optimization
  - Firefly Optimization

Other options

- Build a model of the entire space (Kriging)

- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search
  - Bacterial Colony Optimization
  - Grey Wolf Optimization
  - Firefly Optimization
  - Harmony Search

Other options

- Build a model of the entire space (Kriging)

- Evolutionary algorithms
  - Simulated Annealing
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Particle Swarm
  - Cuckoo Search
  - Bacterial Colony Optimization
  - Grey Wolf Optimization
  - Firefly Optimization
  - Harmony Search
  - River Formation Dynamics

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; f(x)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ **is not available**

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \ \ \text{is not available}$$

$$Ax \leq b$$

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; f(x)$$

$$\textbf{subject to: } x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$$

$$\nabla f(x) \;\; \text{is not available}$$

$$c(x) \leq 0 \;\; (\nabla c(x) \text{ available})$$

Problem Modifications

The Problem:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ **is not available**

$c(x) \leq 0$ **(**$\nabla c(x)$ **unavailable)**

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ is not available

$s(x) \leq 0$ (but $s(x) > 0$ is less desirable)

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ is not available

$x \in \mathbb{Z}$

Problem Modifications

The Problem:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\; f(x) = \tilde{f}(x) + \epsilon \quad \textbf{where} \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ **is not available**

Problem Modifications

The Problem:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\; f(x) = \sum_{i=1}^{m} f_i(x)^2$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f_i(x)$ **is not available**

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; [f_1(x), f_2(x), \ldots, f_m(x)]$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f_i(x)$ **is not available**

Problem Modifications

The Problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

subject to: $x \in \mathcal{D} = \{x : -\infty < l \leq x \leq u < \infty\}$

$\nabla f(x)$ **is not available**

Stochastic

Problem Formulation

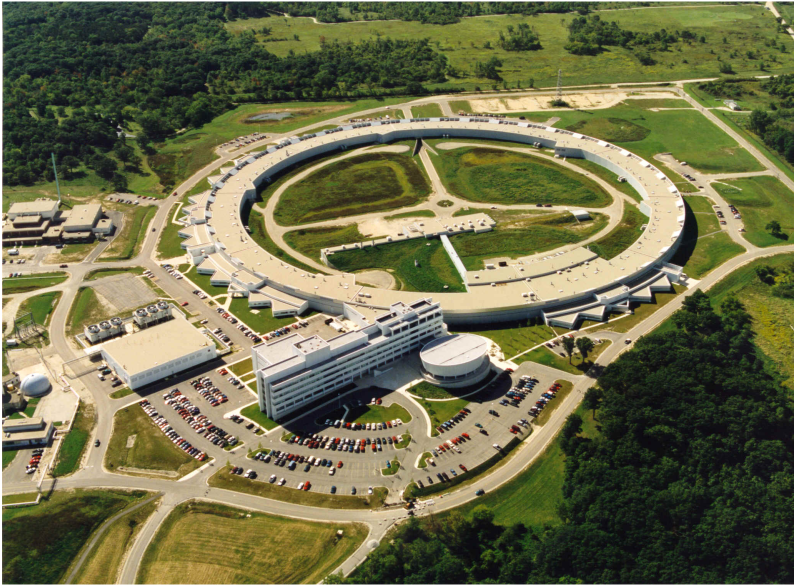
- We want to identify distinct local minimizers of the problem

$$\begin{aligned} \text{minimize}\, f(x) \\ l \le x \le u \\ x \in \mathbb{R}^n \end{aligned}$$

- The simulation f is already using parallel resources, but it does not scale to the entire machine.

- We do not expect hundreds of meaningful local minima.

Motivation



APS: $467M to build; tens of millions to operate.

Multi-Level Single Linkage

A multistart method with some local optimization routine $\mathcal{L}$:

Algorithm 1: MLSL

for $k = 1, 2, \ldots$ **do**

Evaluate f at N random points drawn uniformly from $\mathcal{D}$.

Start $\mathcal{L}$ at any previously evaluated point x:

- that is not a local minima
- $\nexists x_i : \|x - x_i\| \leq r_k$ and $f(x_i) < f(x)$

end

Multi-Level Single Linkage

If the simulation f, and local optimization method $\mathcal{L}$, and r_k satisfy some assumptions, then MLSL has nice theoretical properties.

- f
  - Twice continuously differentiable
  - All local minima are interior points
  - Some positive distance between all minima

Multi-Level Single Linkage

If the simulation f, and local optimization method $\mathcal{L}$, and r_k satisfy some assumptions, then MLSL has nice theoretical properties.

- f
  - Twice continuously differentiable
  - All local minima are interior points
  - Some positive distance between all minima
- $\mathcal{L}$
  - Strictly descent
  - Converges to nearby minimum (not stationary point)

Multi-Level Single Linkage

If the simulation f, and local optimization method $\mathcal{L}$, and r_k satisfy some assumptions, then MLSL has nice theoretical properties.

- f
  - Twice continuously differentiable
  - All local minima are interior points
  - Some positive distance between all minima
- $\mathcal{L}$
  - Strictly descent
  - Converges to nearby minimum (not stationary point)
- r_k

$$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\Gamma\left(1 + \frac{n}{2}\right) \mathrm{vol}\left(\mathcal{D}\right) \frac{\sigma \log kN}{kN}} \tag{1}$$

Multi-Level Single Linkage

Theorem 8[1]

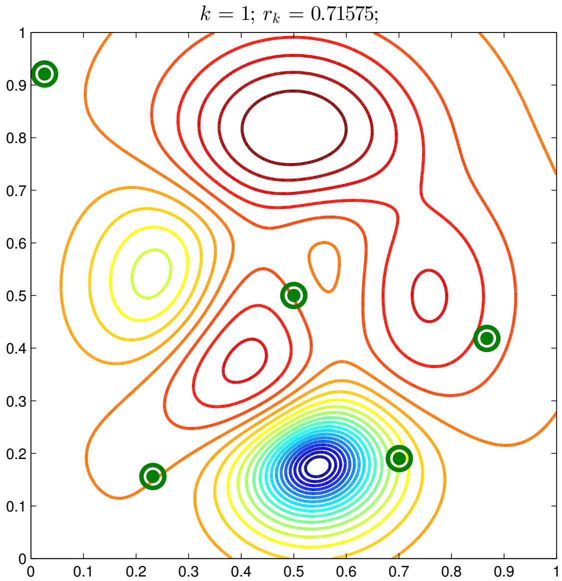
If r_k is defined by (1) and $\sigma > 2$, then the probability that $\mathcal{L}$ is started at iteration k tends to 0 with increasing k.
If $\sigma > 4$, then, even if the sampling continues forever, the total number of local searches ever started is finite with probability 1.

Theorem 12[1]

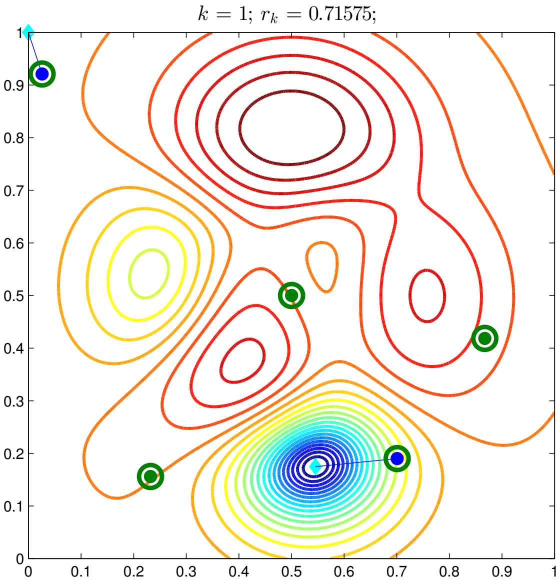
If $r_k \to 0$, all local minima will be found.

[1] A. H. G. Rinnooy Kan and G. T. Timmer. "Stochastic Global Optimization Methods Part II: Multi Level Methods." *Mathematical Programming*, 39(1):57–78, Sept. 1987.

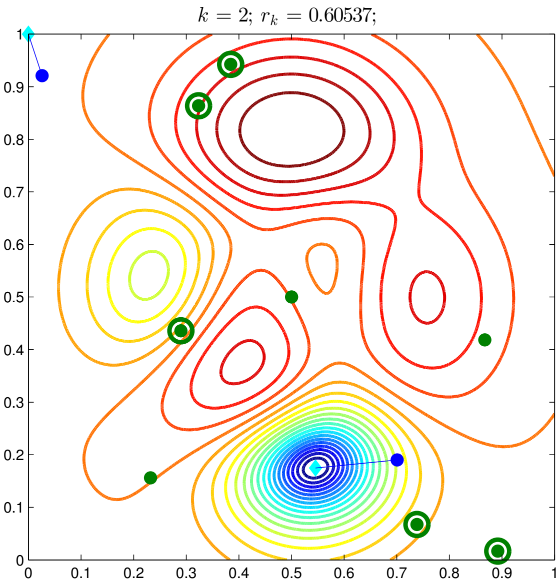
Multi-Level Single Linkage



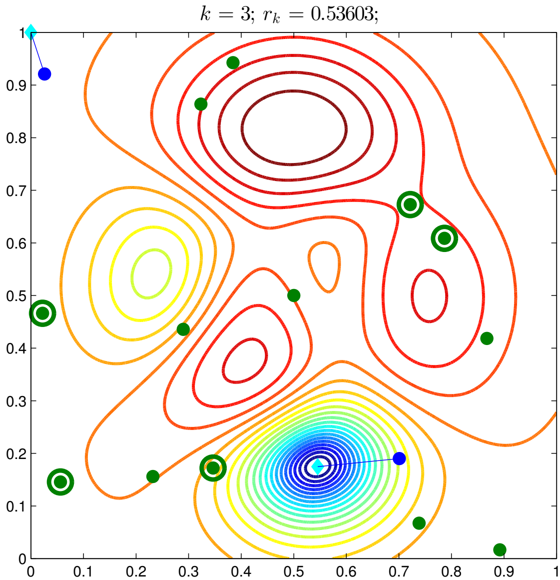
Multi-Level Single Linkage



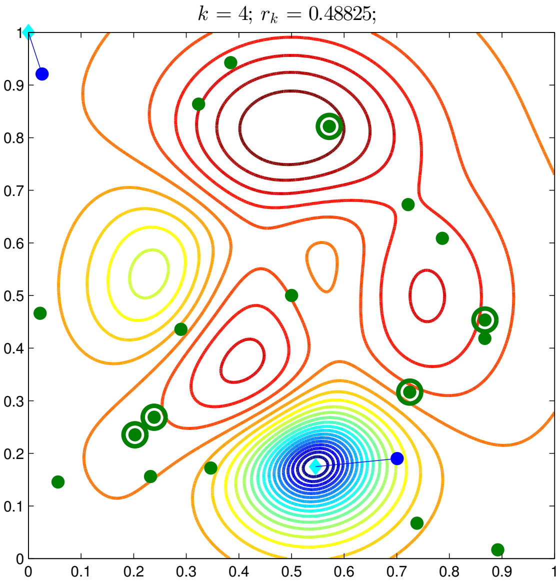
Multi-Level Single Linkage



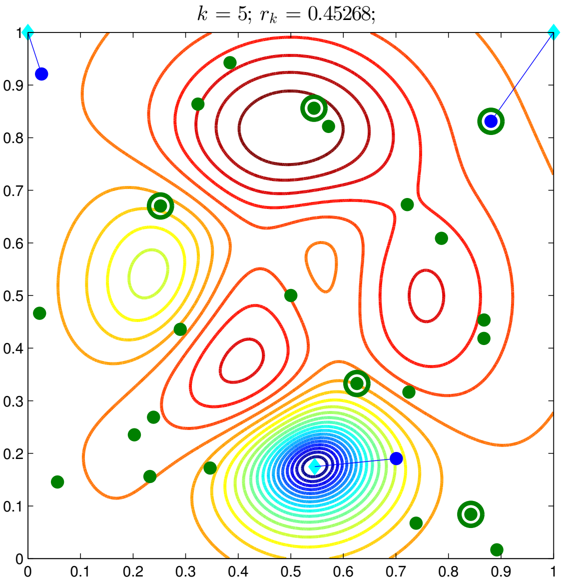
Multi-Level Single Linkage



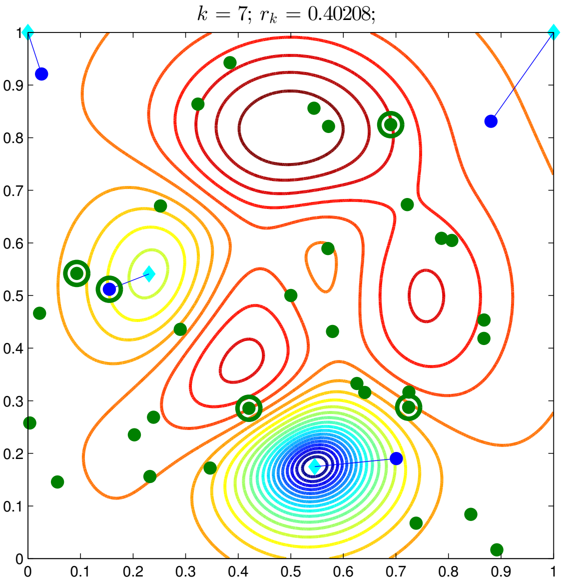
Multi-Level Single Linkage



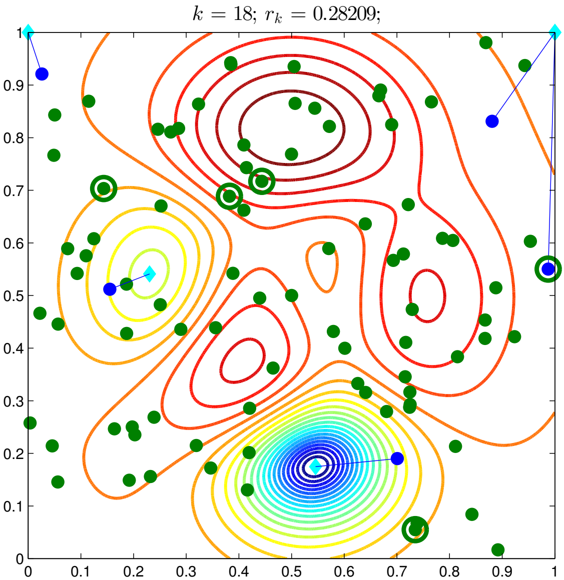
Multi-Level Single Linkage



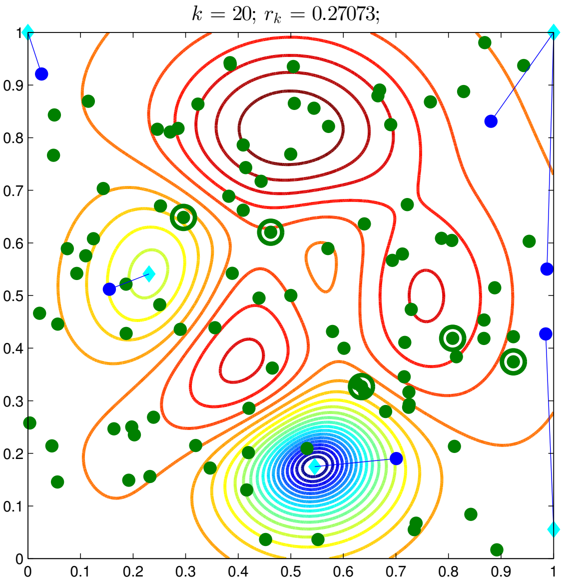
Multi-Level Single Linkage



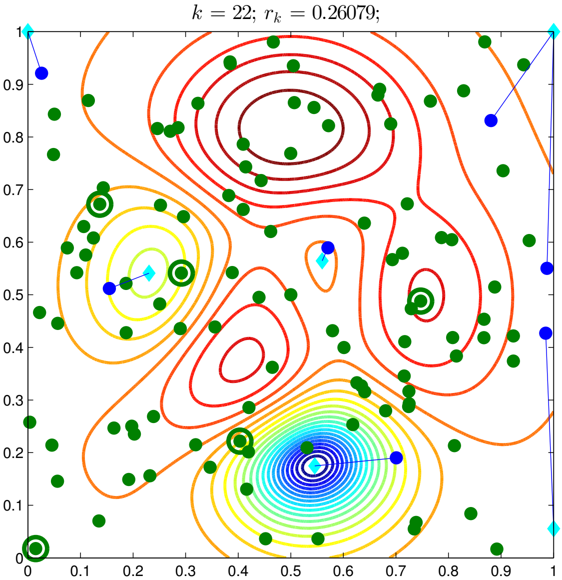
Multi-Level Single Linkage



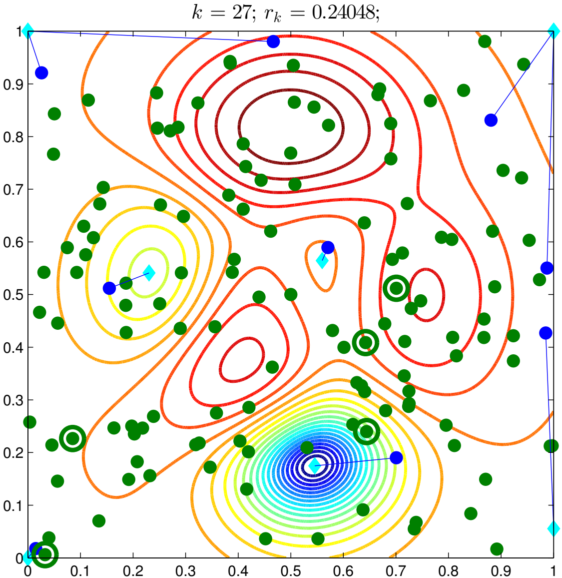
Multi-Level Single Linkage



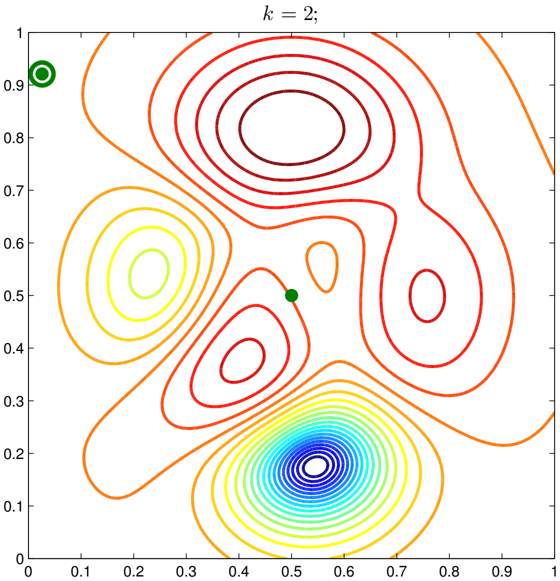
Multi-Level Single Linkage



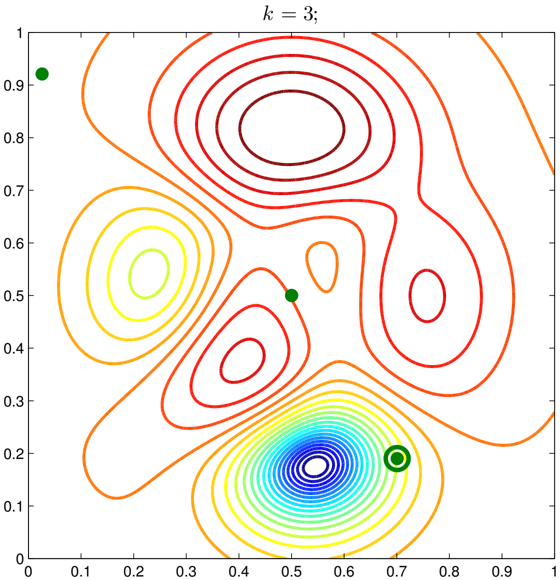
Multi-Level Single Linkage



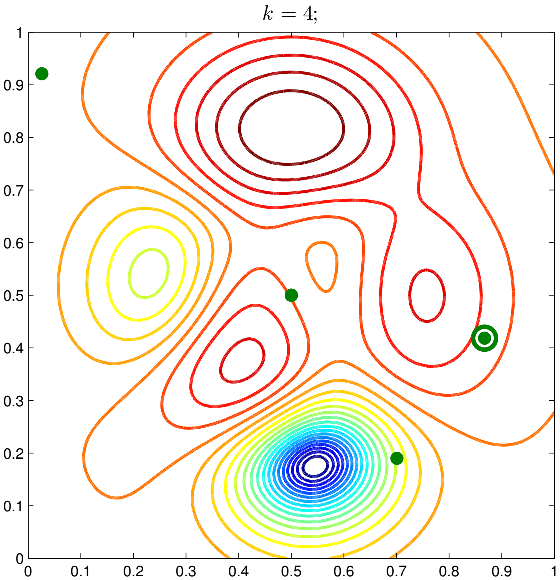
Derivative-free MLSL



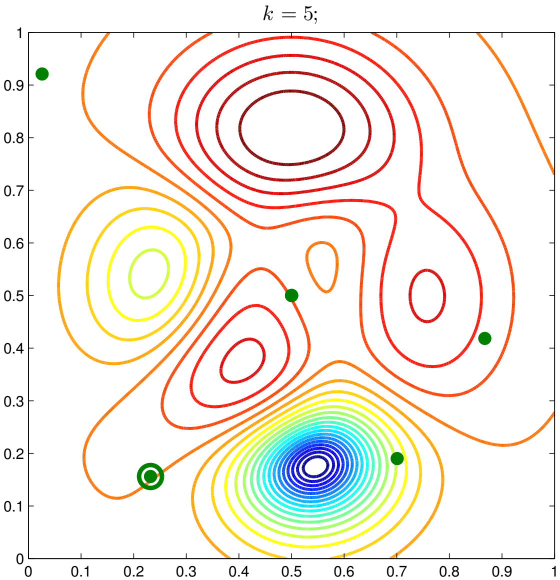
Derivative-free MLSL



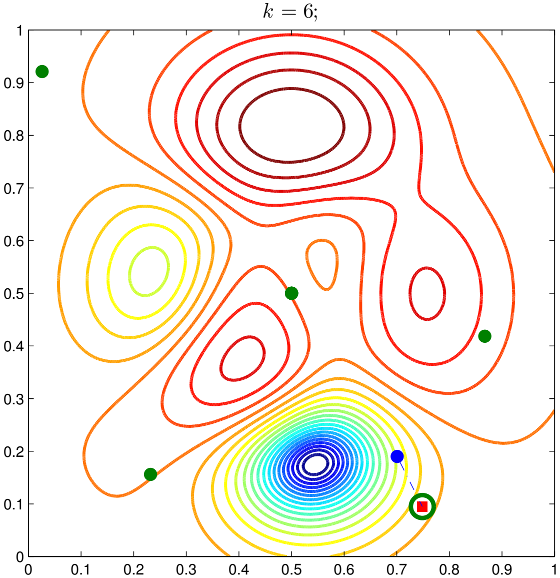
Derivative-free MLSL



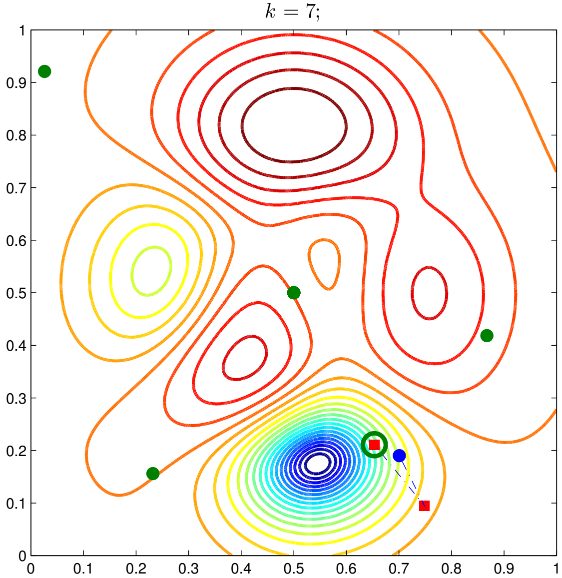
Derivative-free MLSL



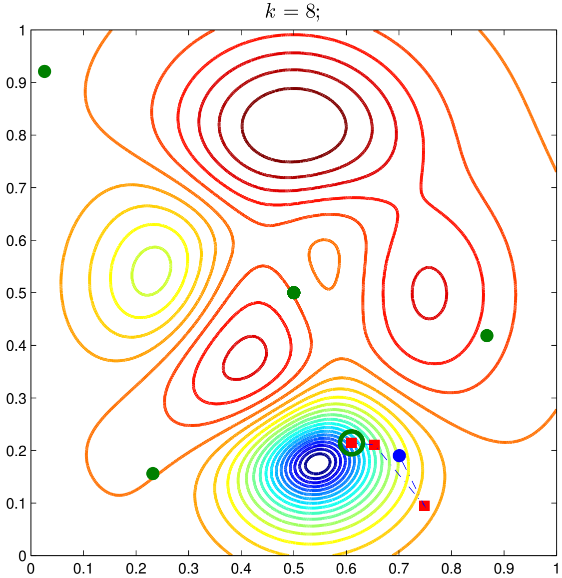
Derivative-free MLSL



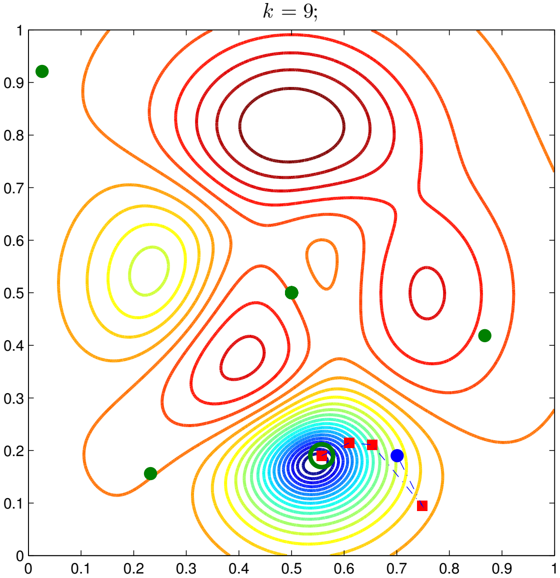
Derivative-free MLSL



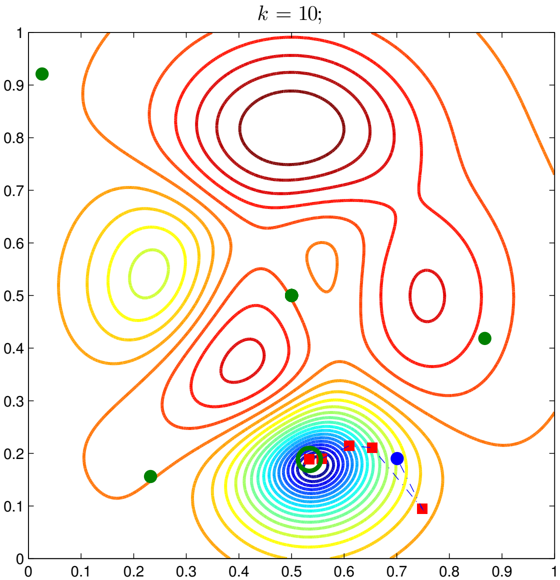
Derivative-free MLSL



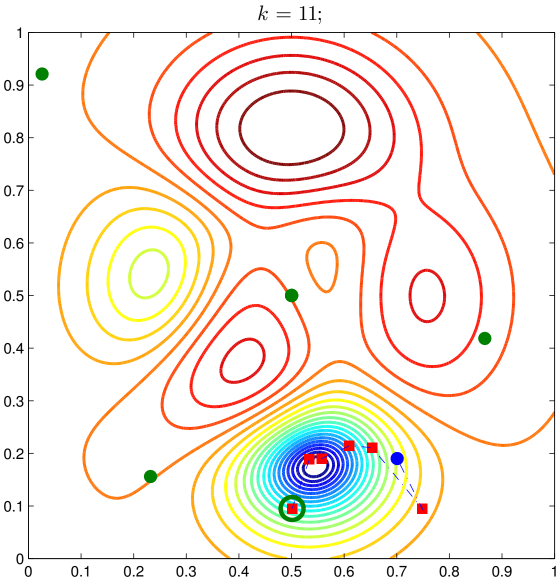
Derivative-free MLSL



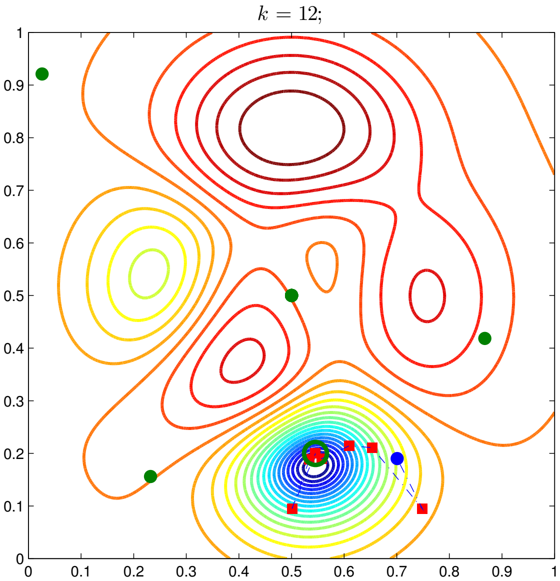
Derivative-free MLSL



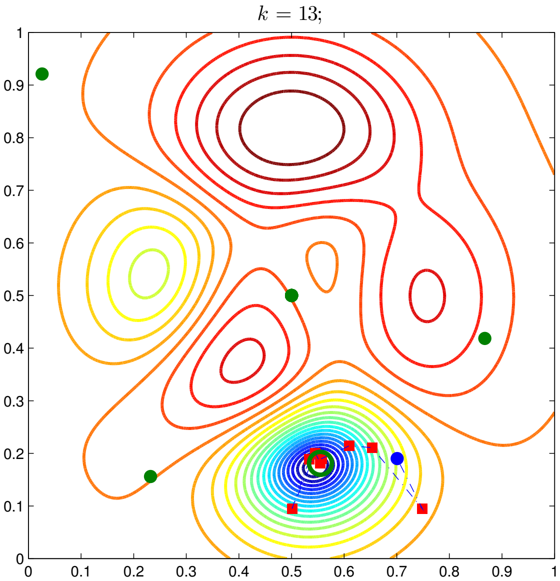
Derivative-free MLSL



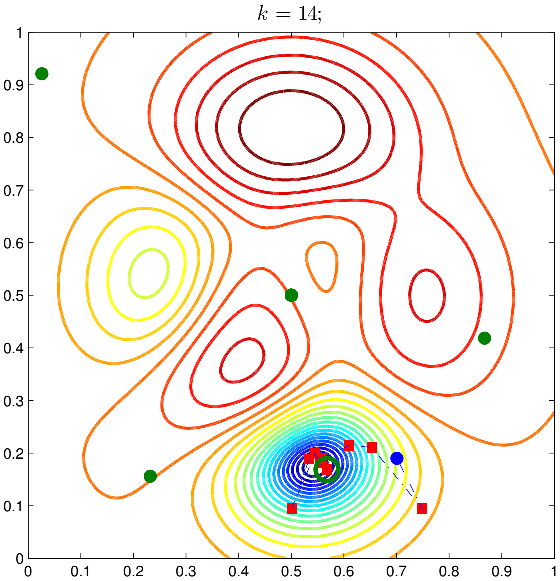
Derivative-free MLSL



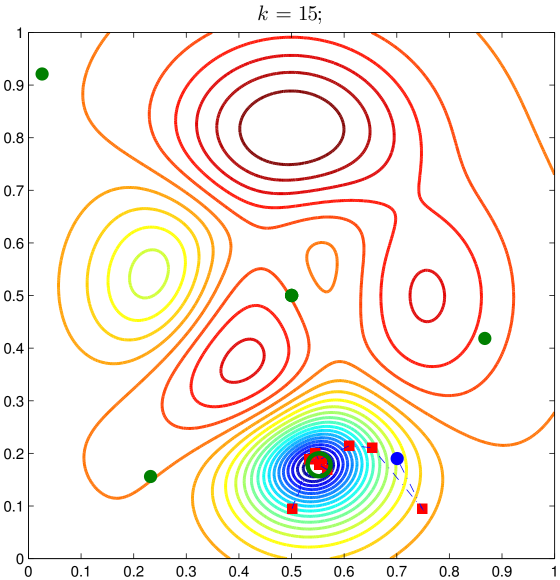
Derivative-free MLSL



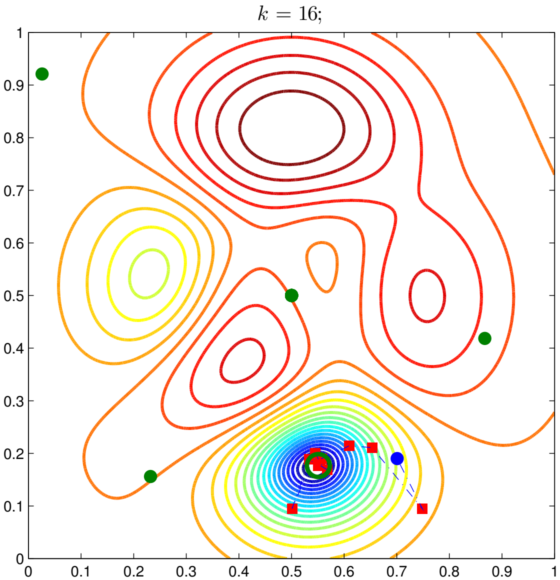
Derivative-free MLSL



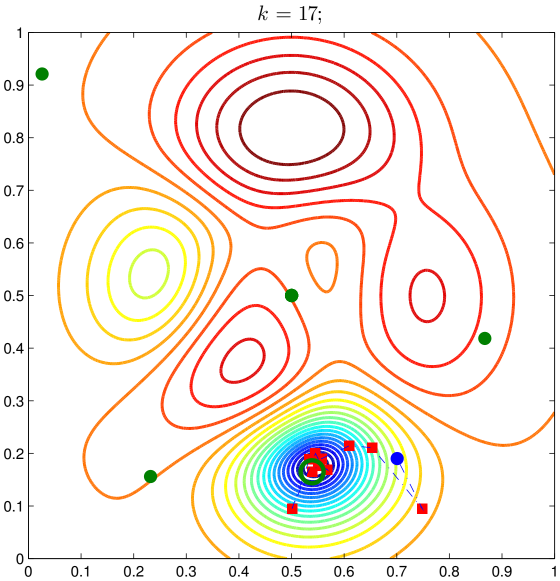
Derivative-free MLSL



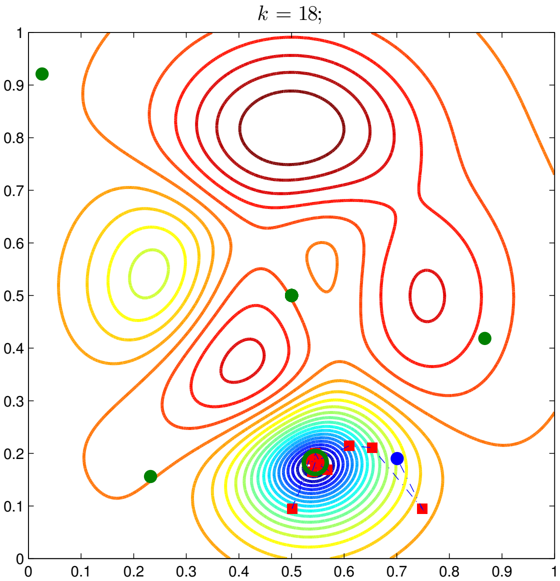
Derivative-free MLSL



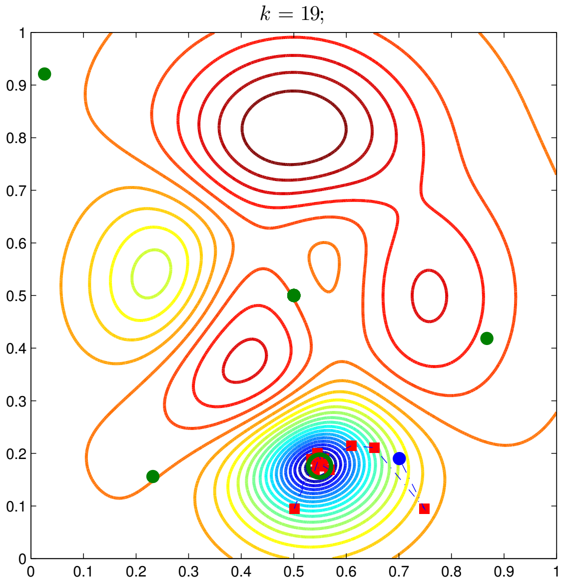
Derivative-free MLSL



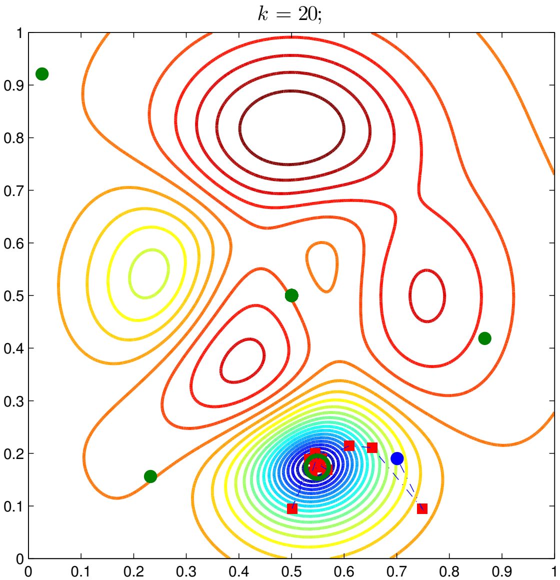
Derivative-free MLSL



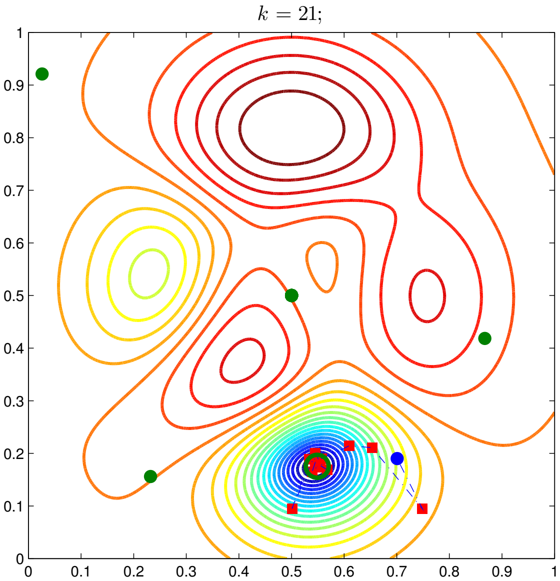
Derivative-free MLSL



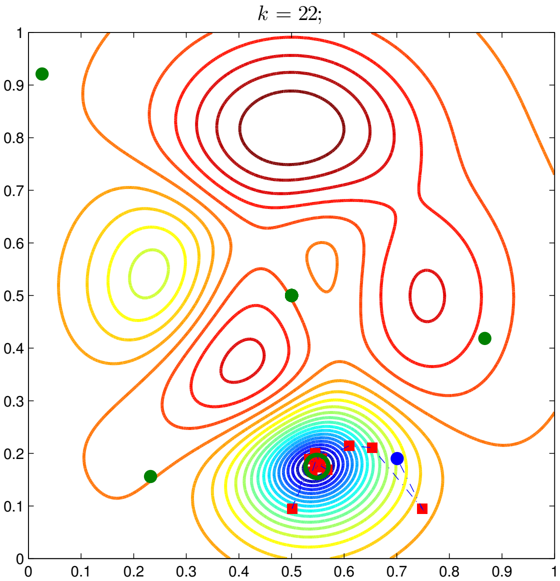
Derivative-free MLSL



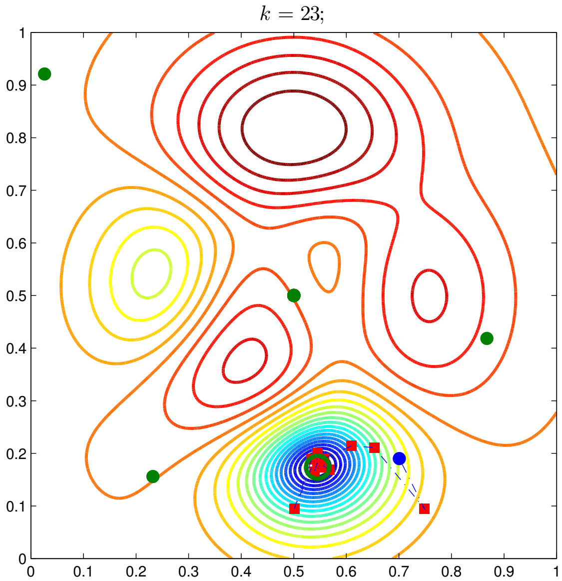
Derivative-free MLSL



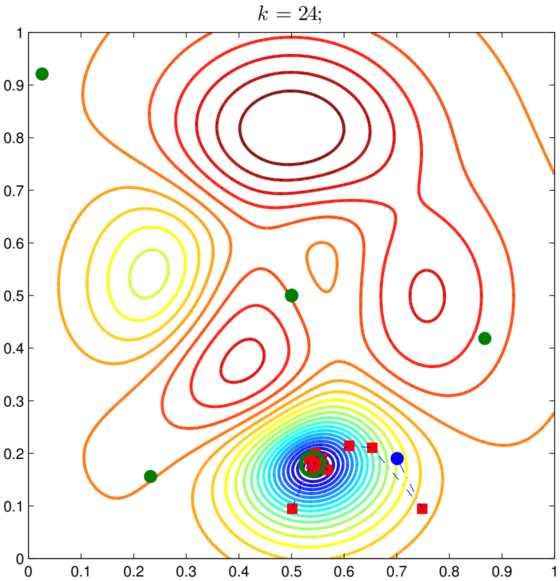
Derivative-free MLSL



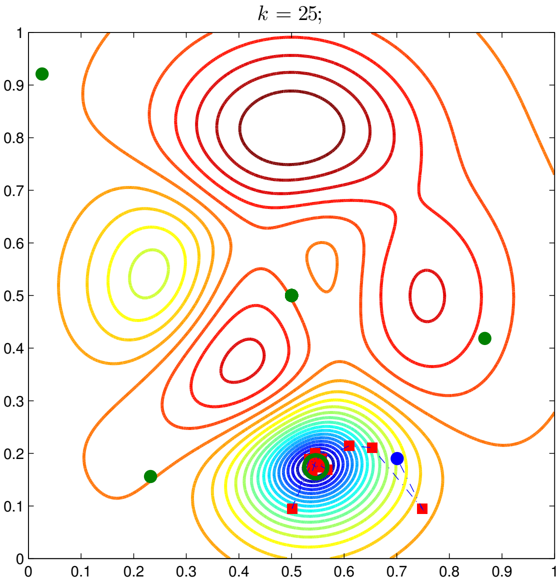
Derivative-free MLSL



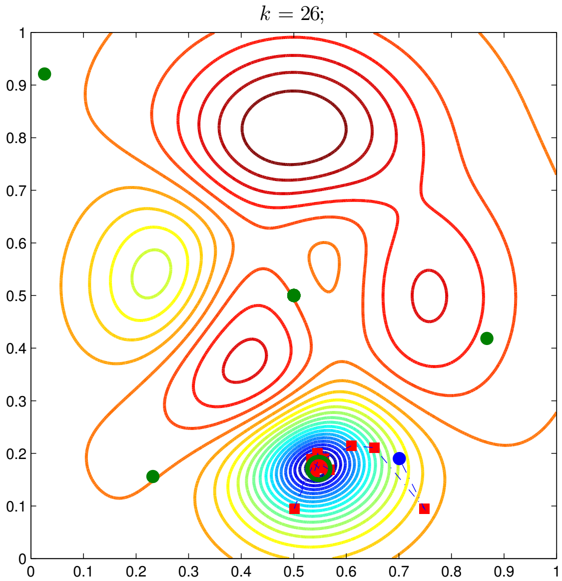
Derivative-free MLSL



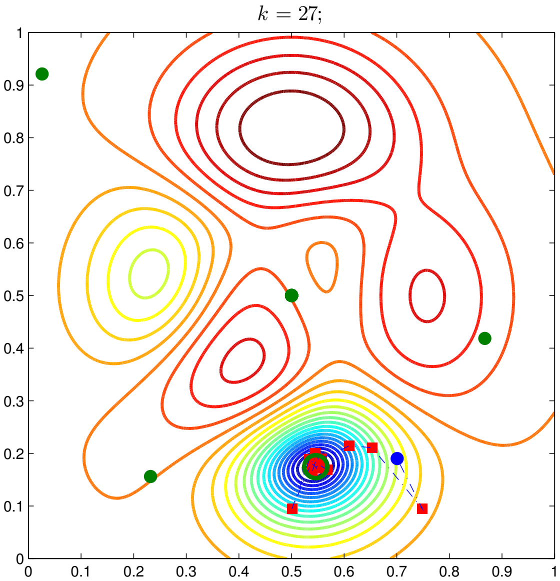
Derivative-free MLSL



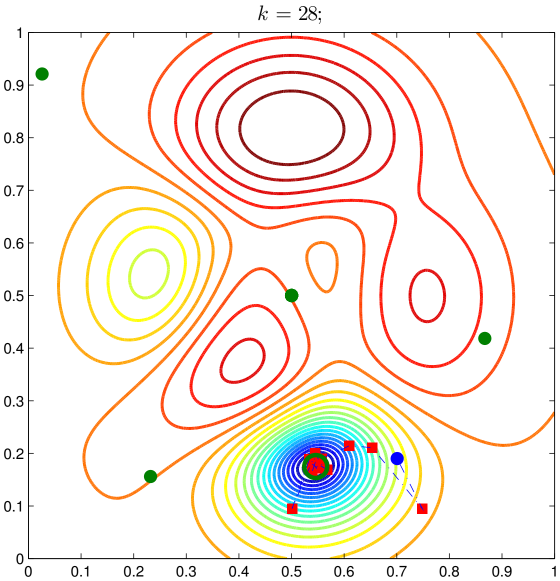
Derivative-free MLSL



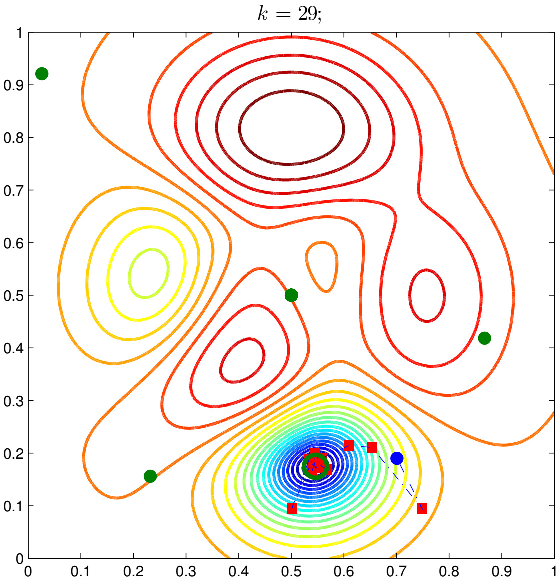
Derivative-free MLSL



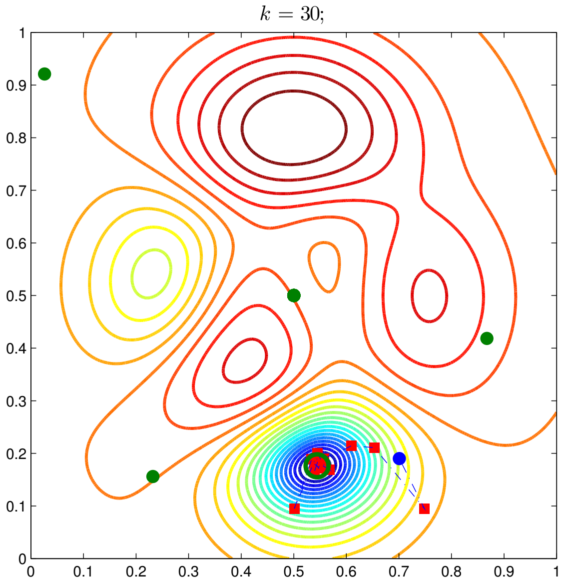
Derivative-free MLSL



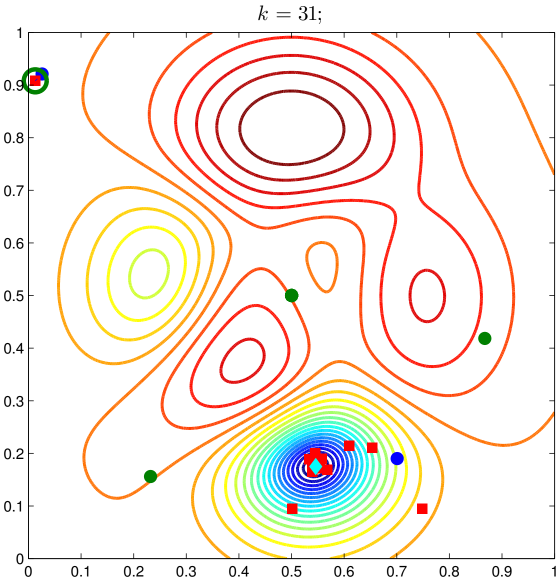
Derivative-free MLSL



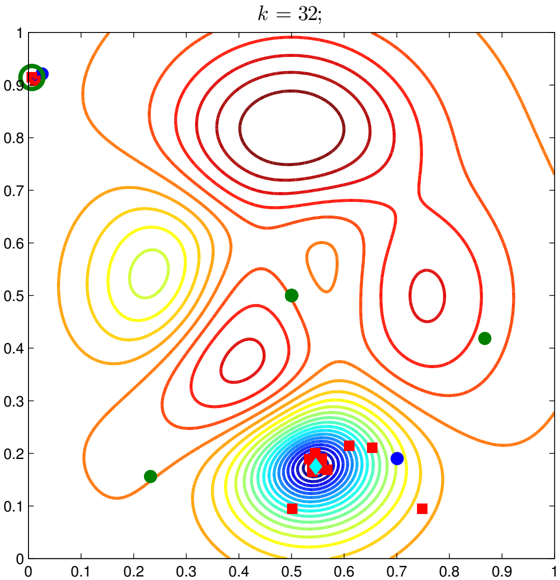
Derivative-free MLSL



Derivative-free MLSL



Derivative-free MLSL

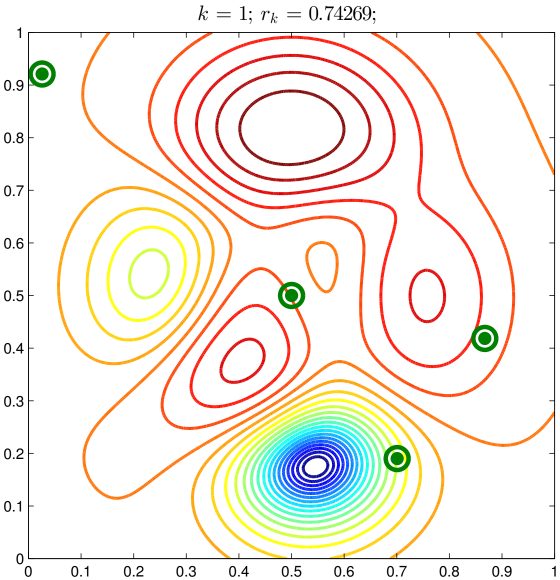


Derivative-free MLSL

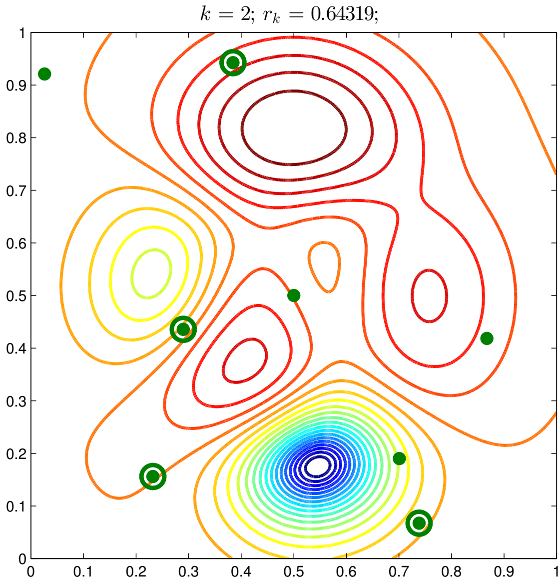
- If we had the ability to concurrently evaluate points, what might be the best course?

- Is it better to do finite-difference computations or separate local optimization runs?

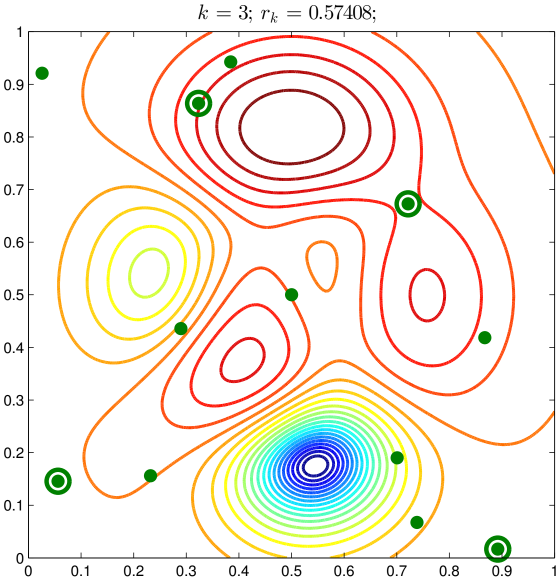
Our Method



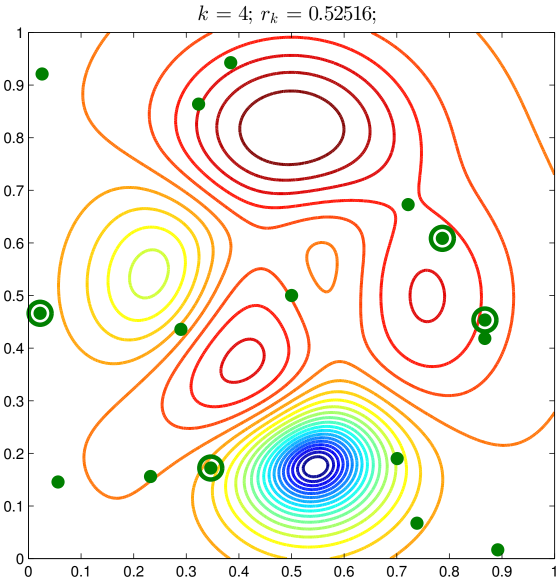
Our Method



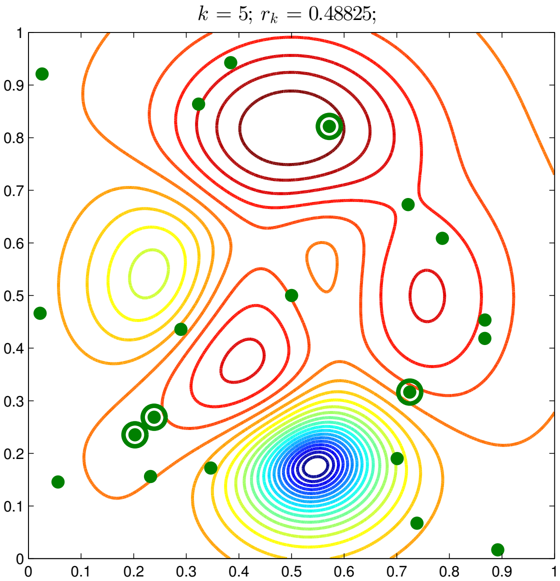
Our Method



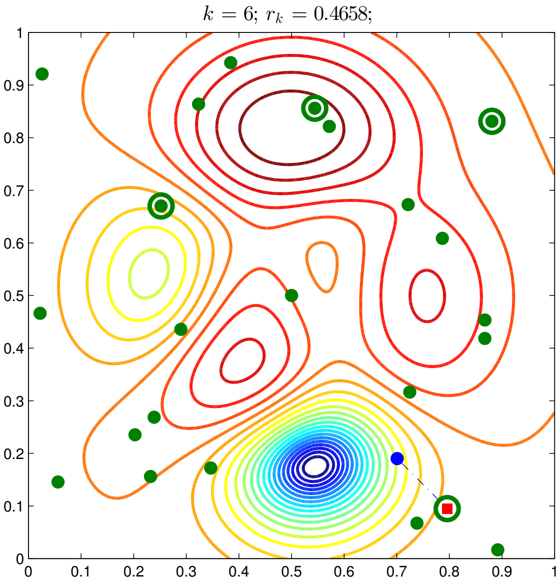
Our Method



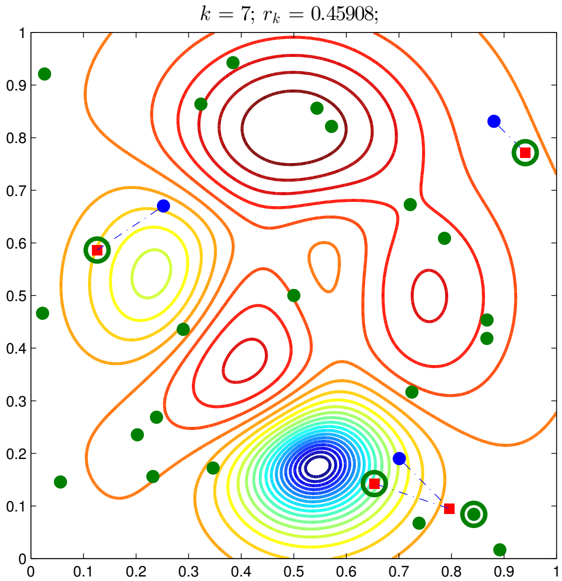
Our Method



Our Method

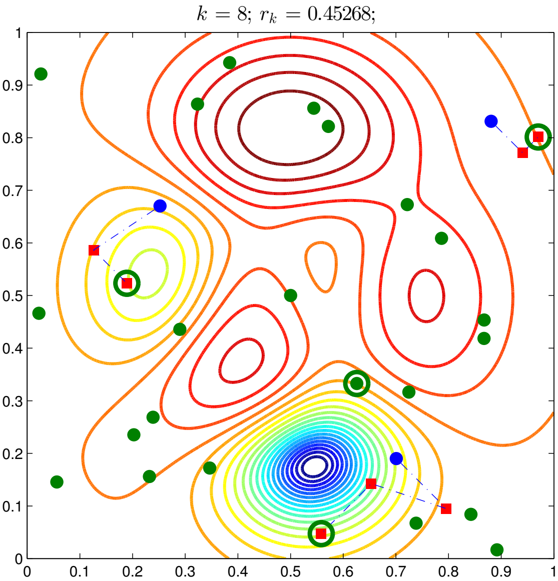


Our Method

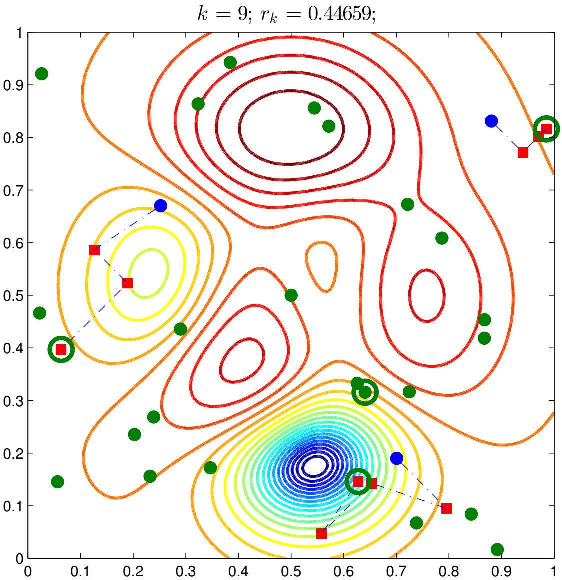


$k = 7;\ r_k = 0.45908;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

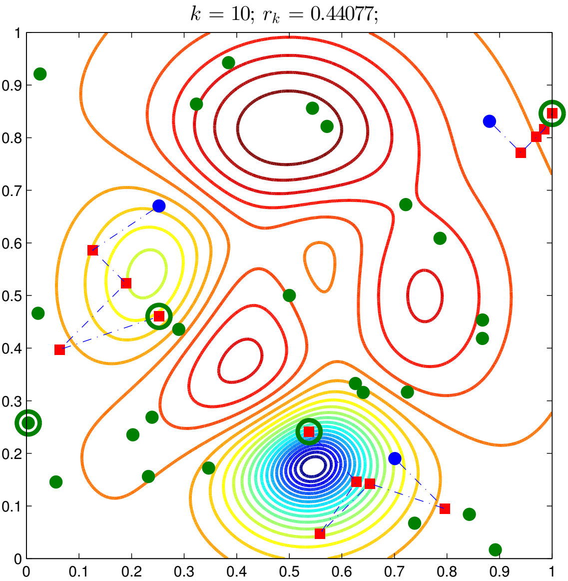
Our Method



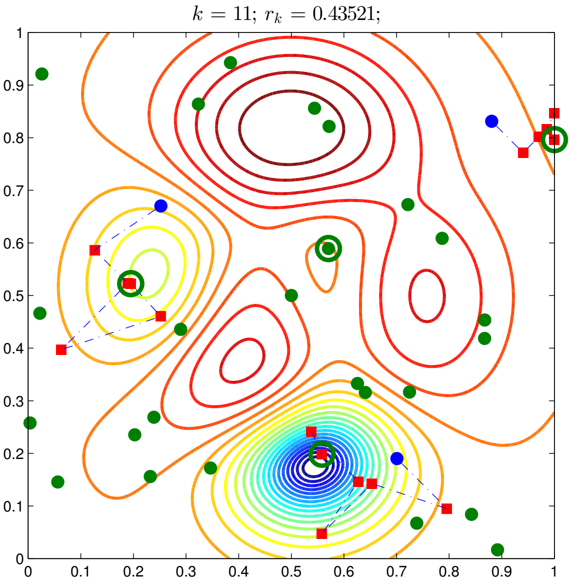
Our Method



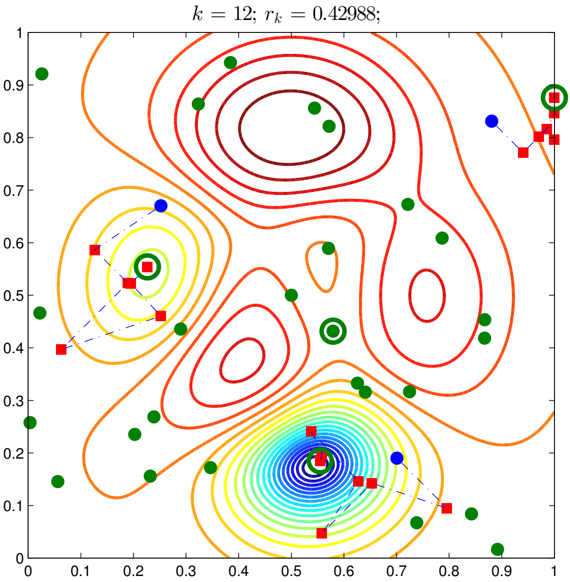
Our Method



Our Method

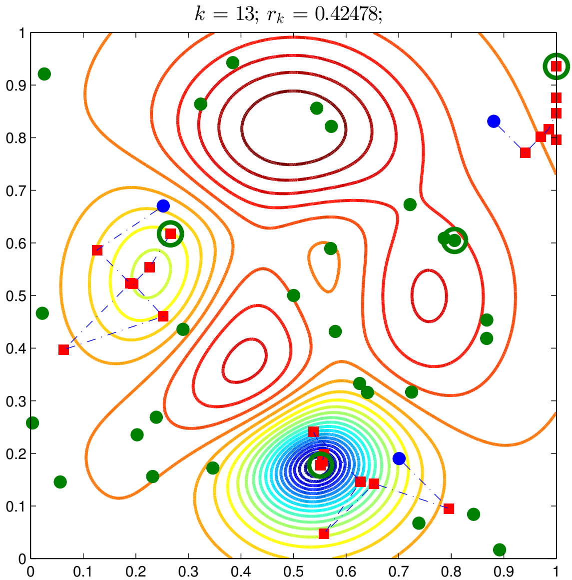


Our Method

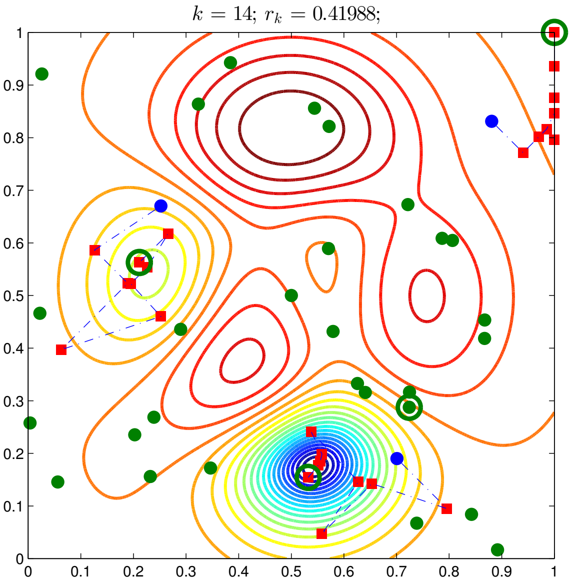


$k = 12;\ r_k = 0.42988;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

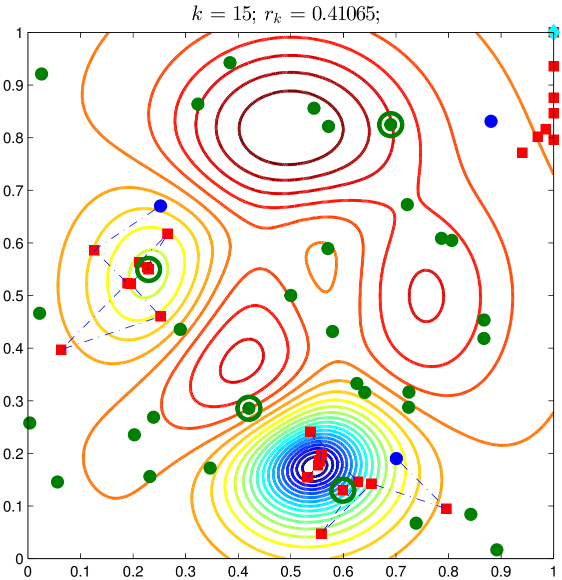
Our Method



Our Method

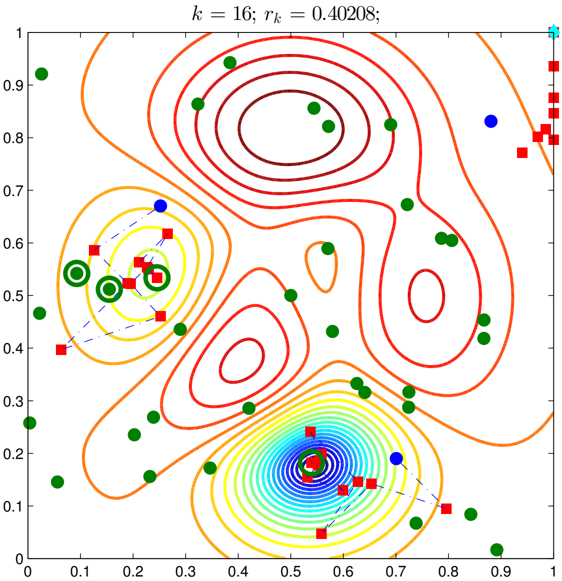


Our Method

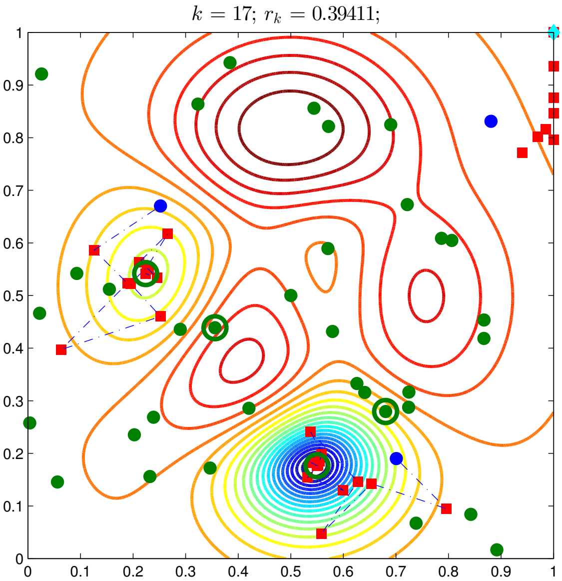


$k = 15;\ r_k = 0.41065;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

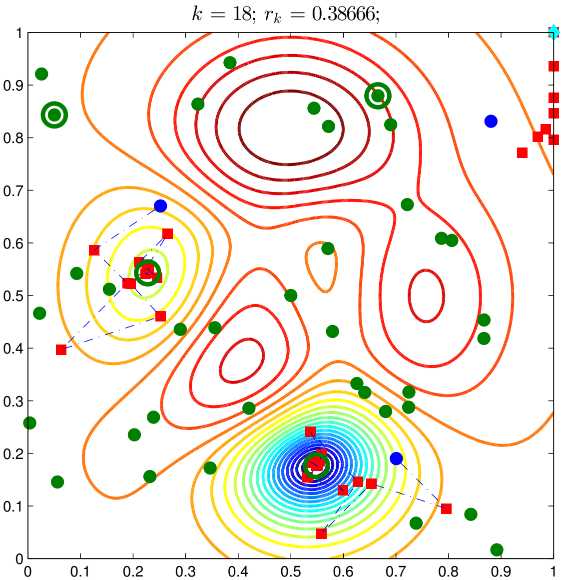
Our Method



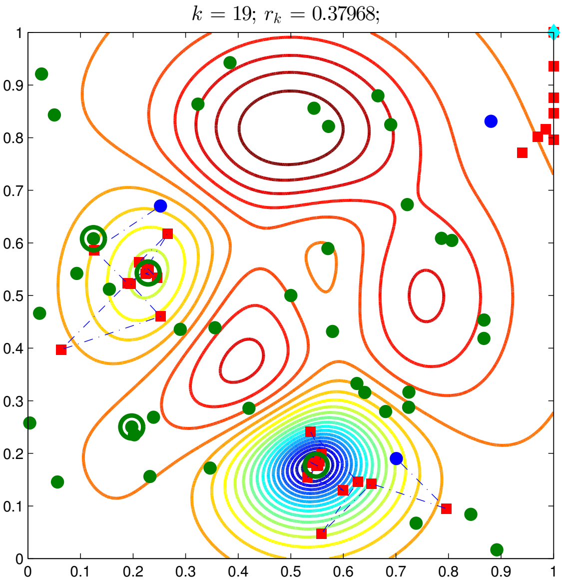
Our Method



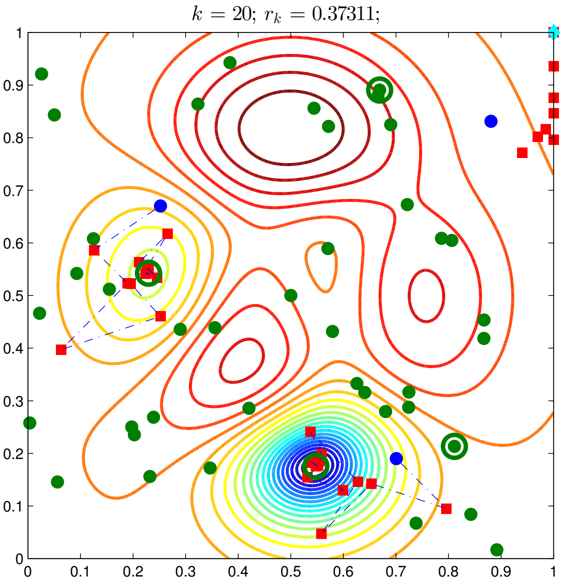
Our Method



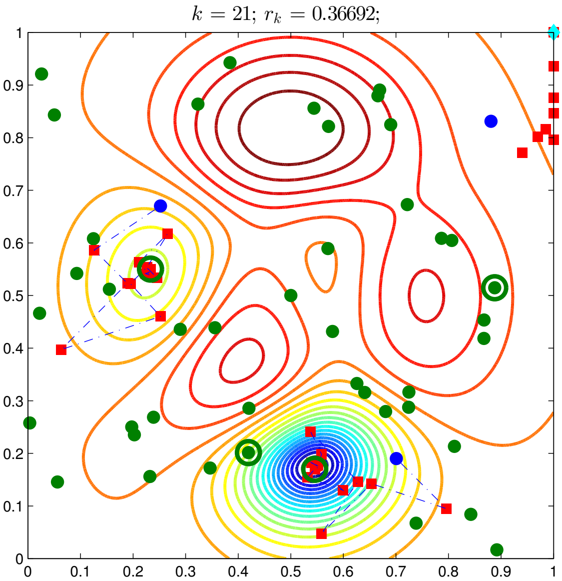
Our Method



Our Method

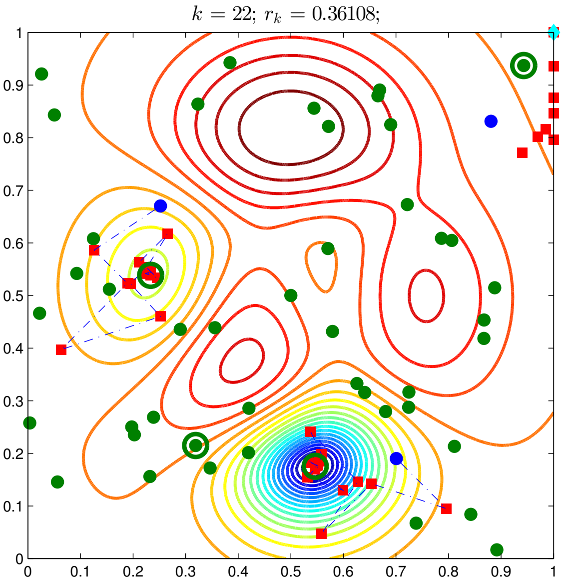


Our Method

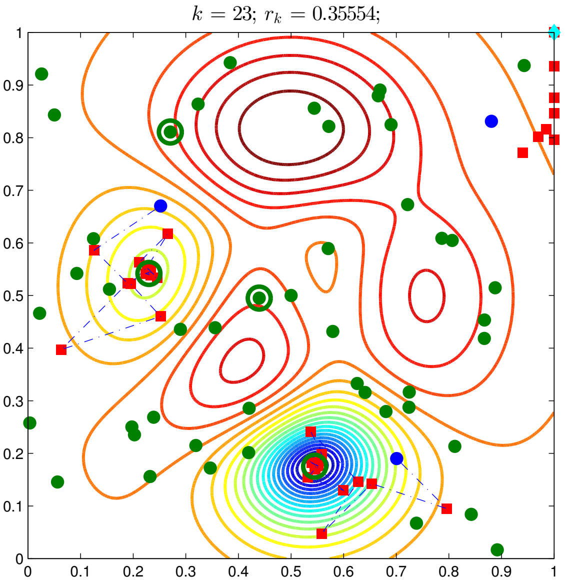


$k = 21;\ r_k = 0.36692;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

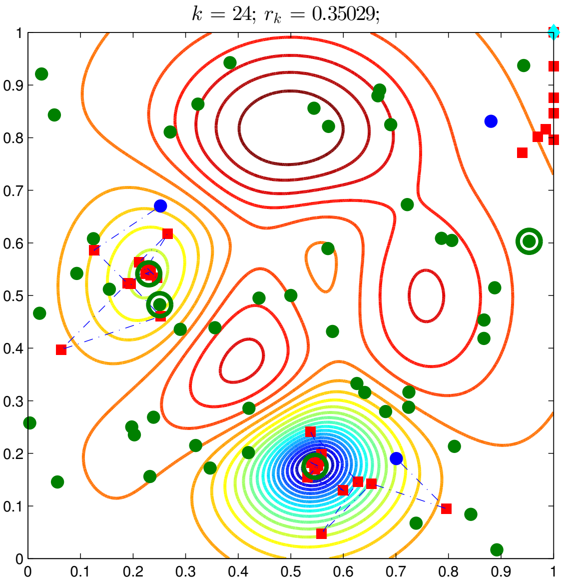
Our Method



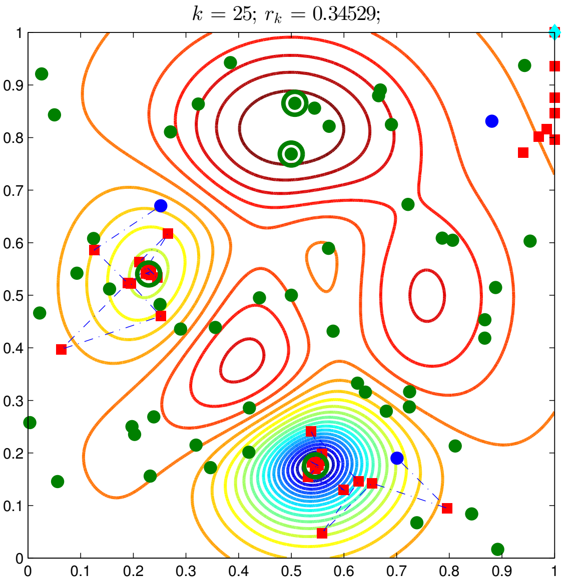
Our Method



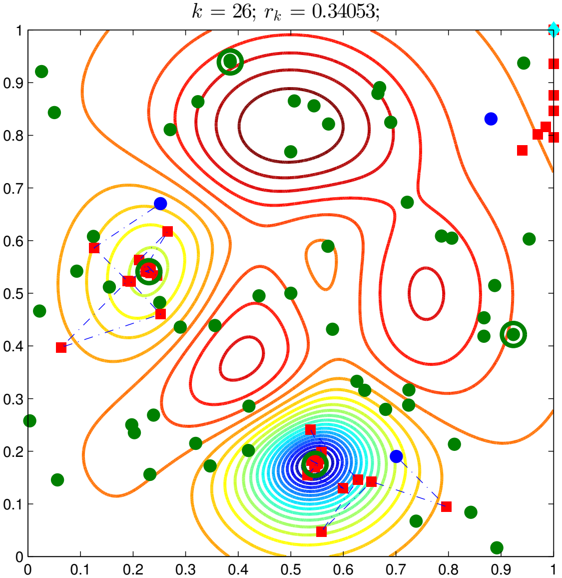
Our Method



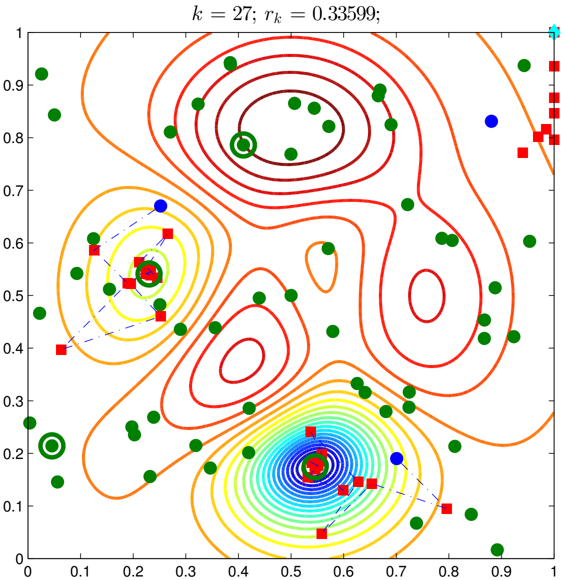
Our Method



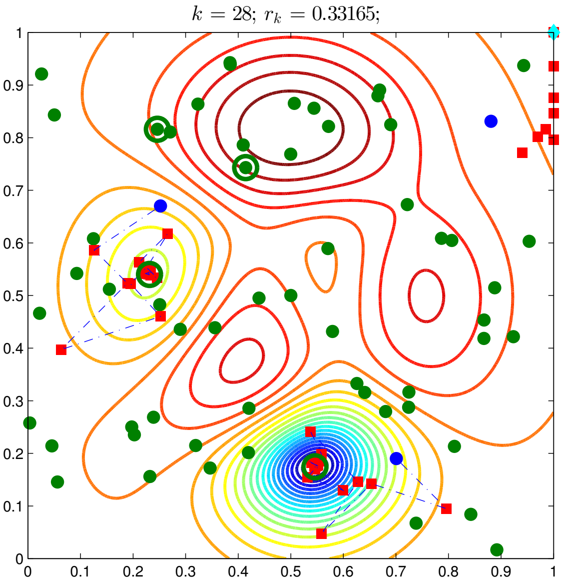
Our Method



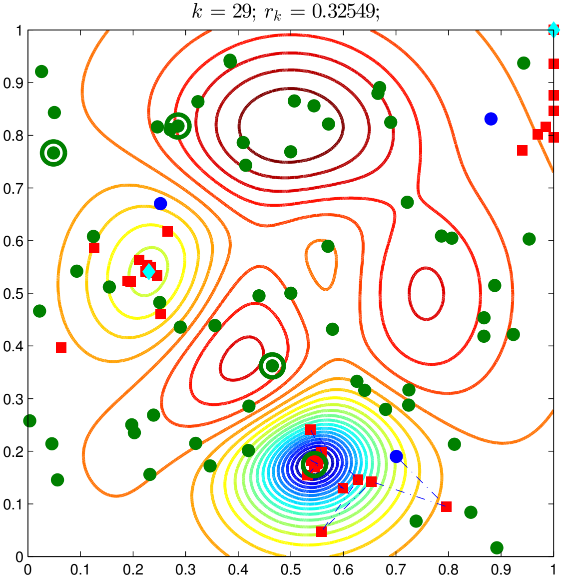
Our Method



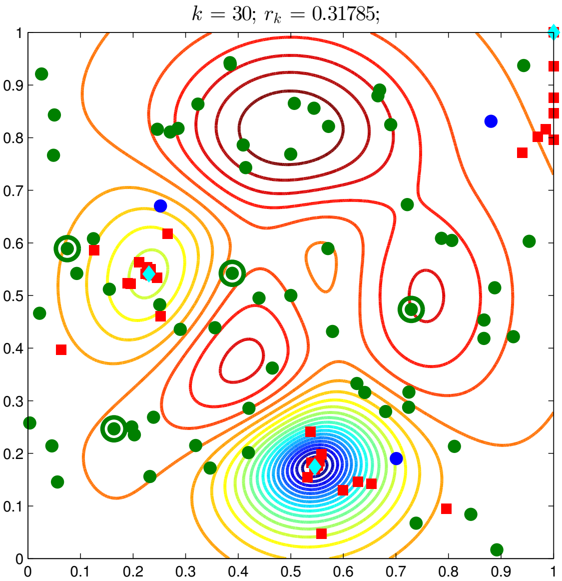
Our Method



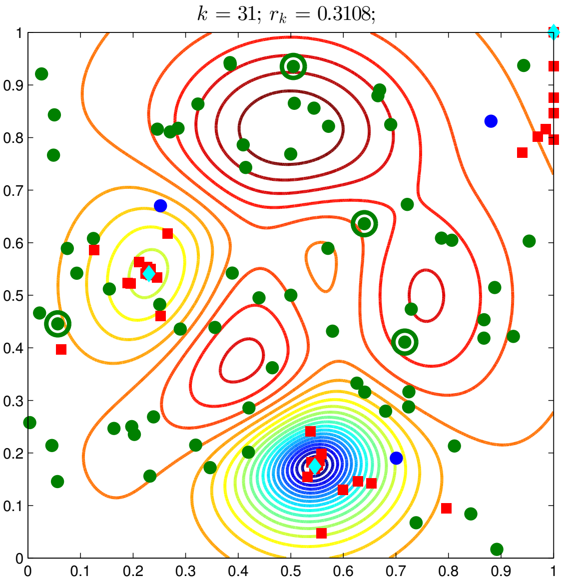
Our Method



Our Method

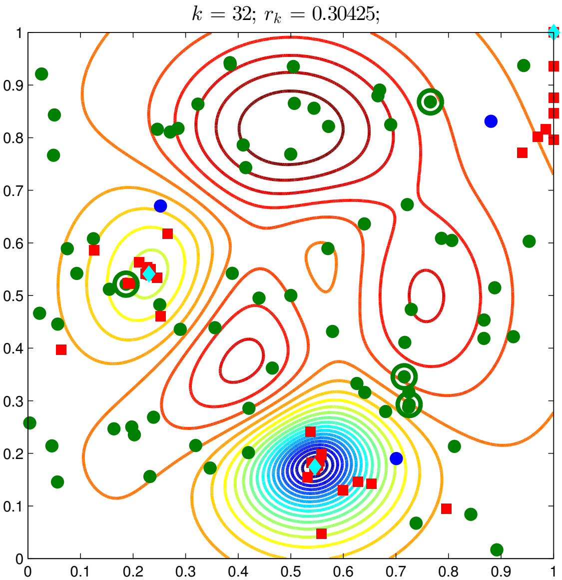


Our Method

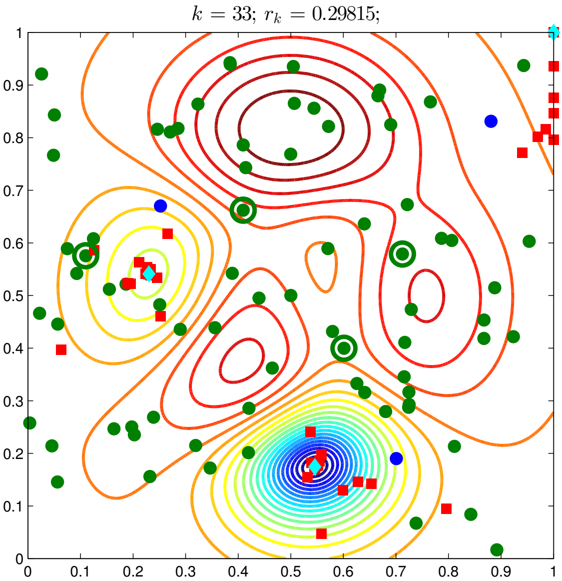


$k = 31;\ r_k = 0.3108;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Our Method

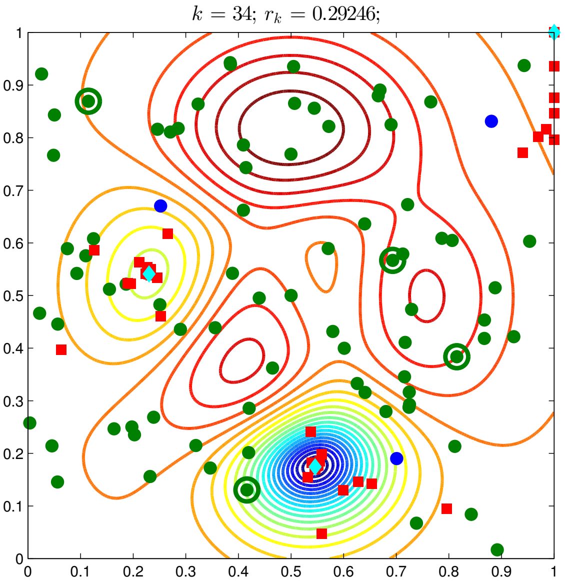


Our Method

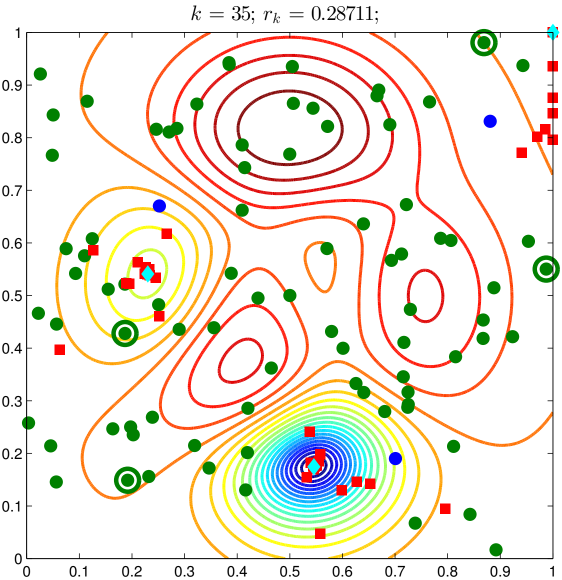


$k = 33$; $r_k = 0.29815$;
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Our Method

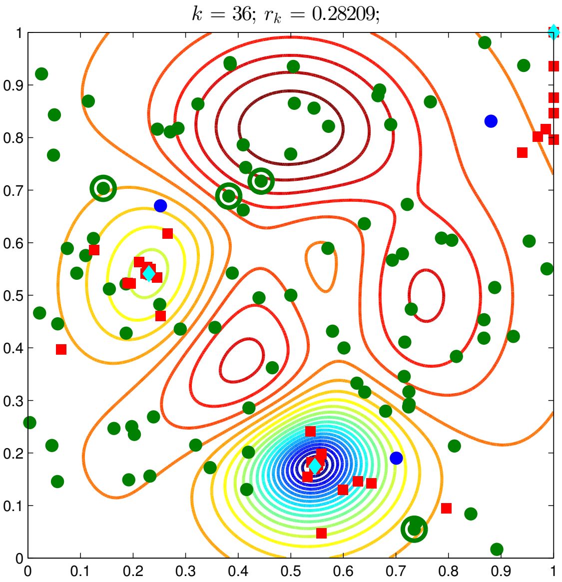


Our Method

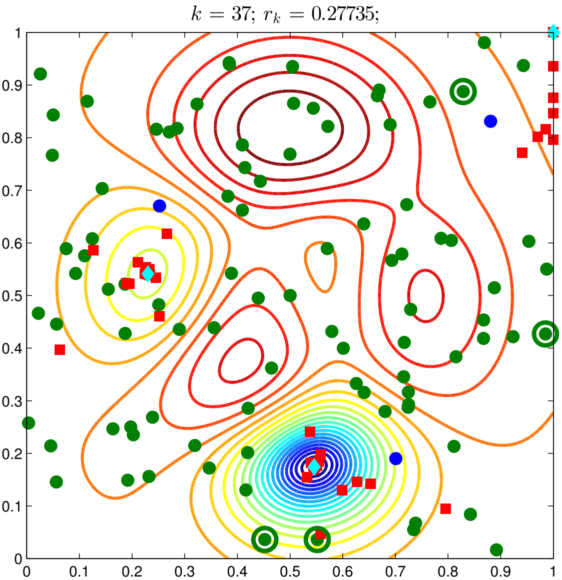


$k = 35; r_k = 0.28711;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

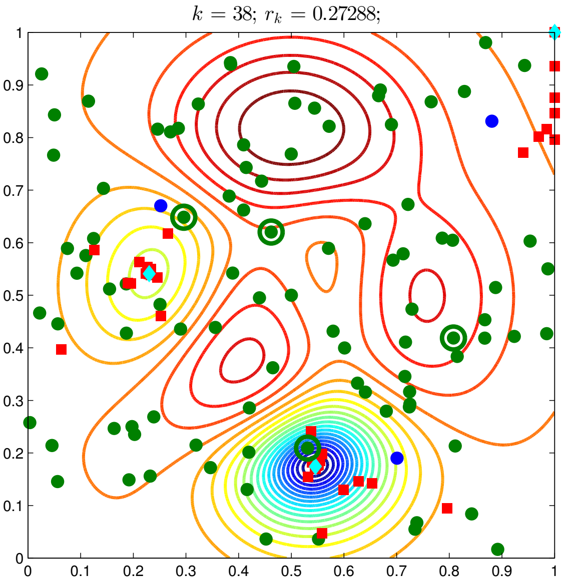
Our Method



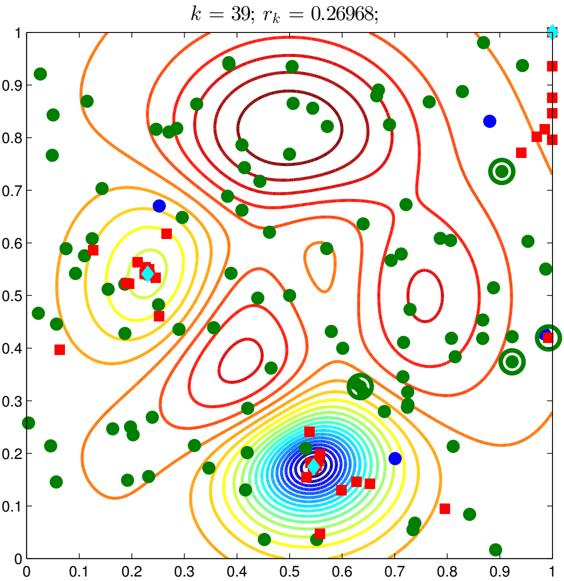
Our Method



Our Method

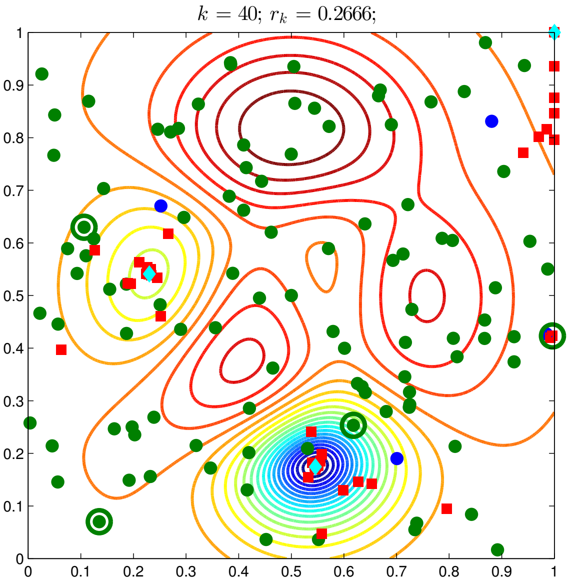


Our Method

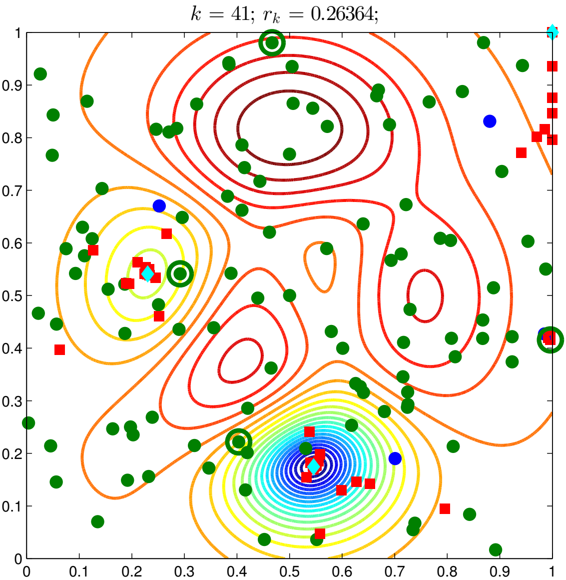


$k = 39;\ r_k = 0.26968;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

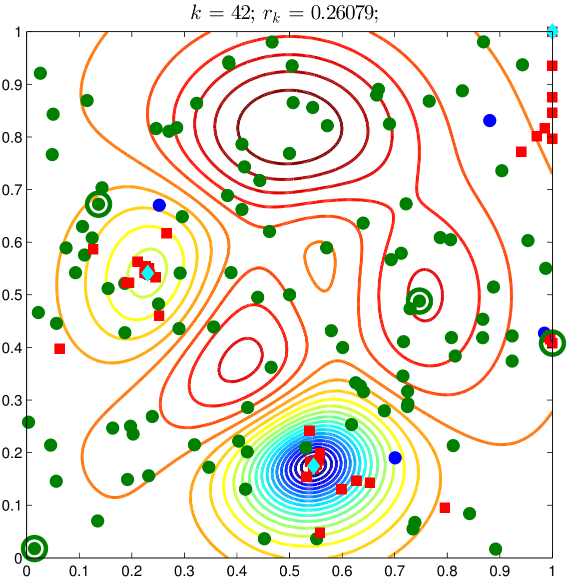
Our Method



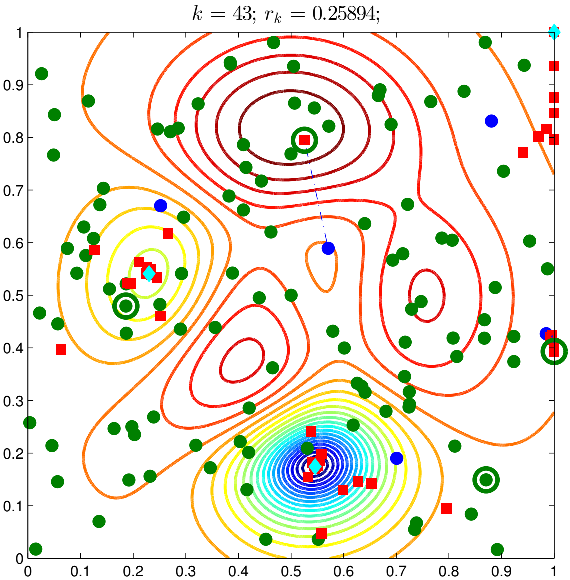
Our Method



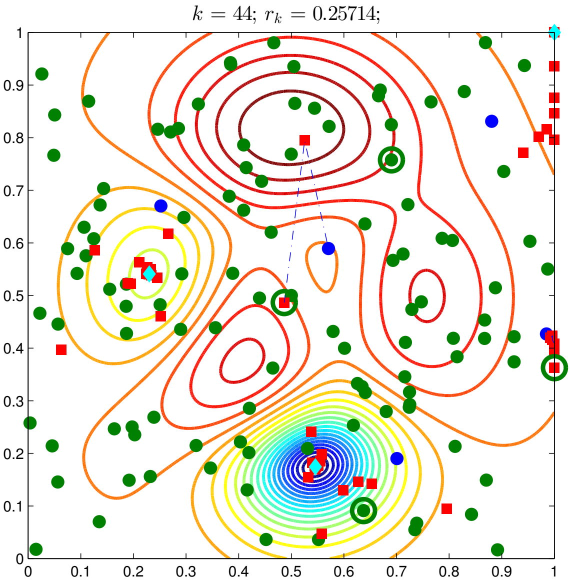
Our Method



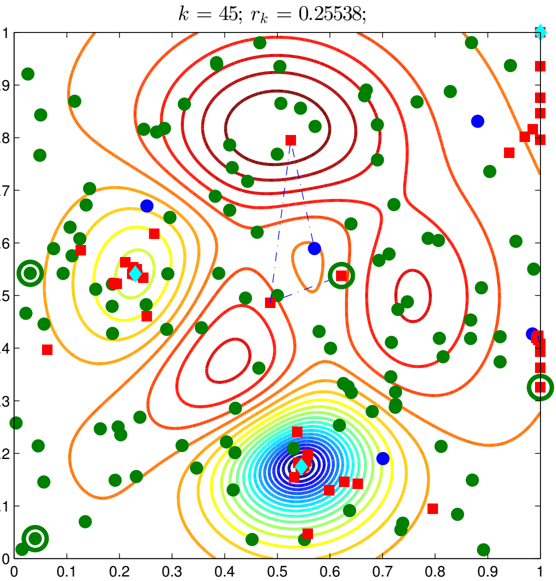
Our Method



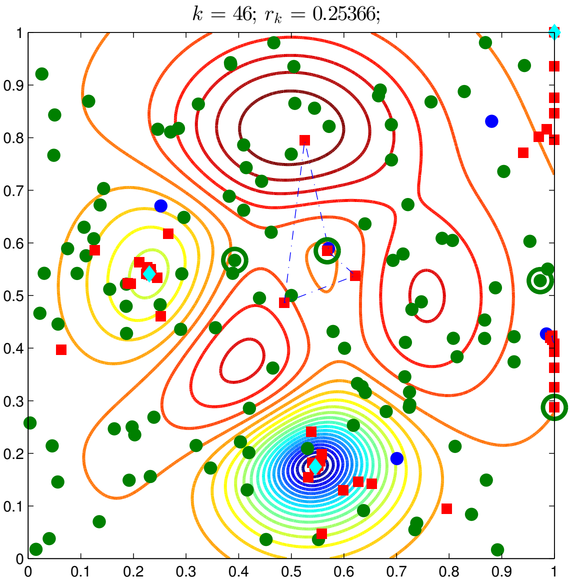
Our Method



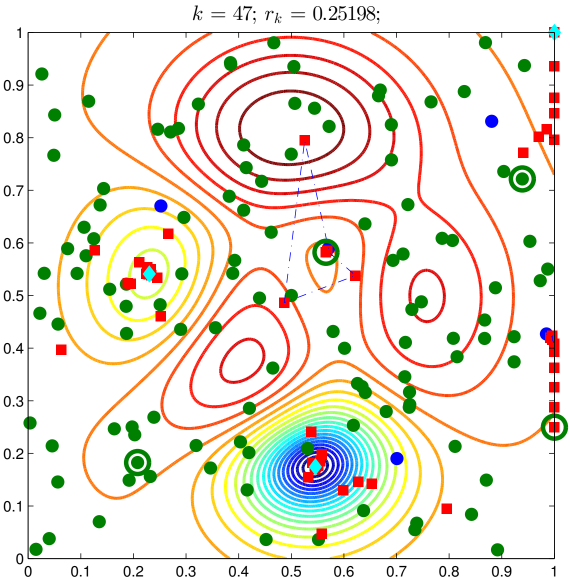
Our Method



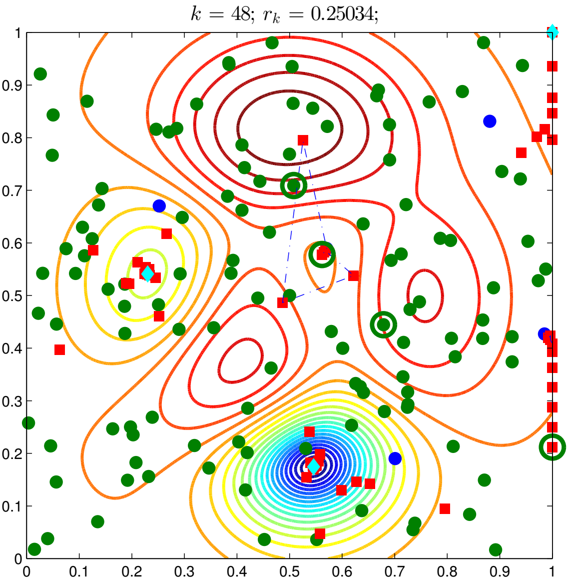
Our Method



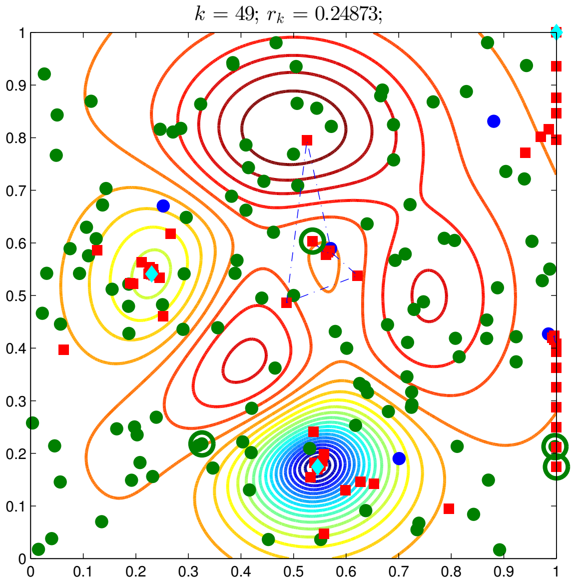
Our Method



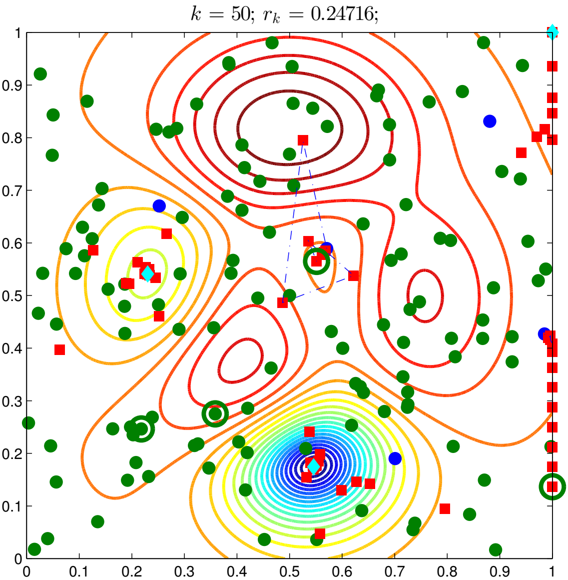
Our Method



Our Method

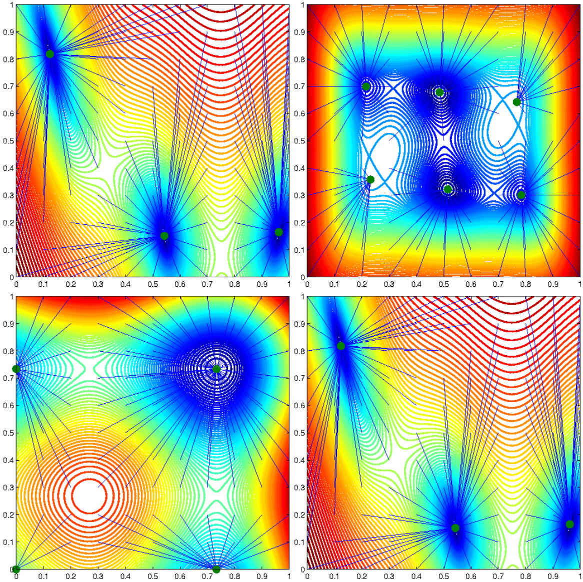


Our Method



$k = 50;\ r_k = 0.24716;$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Measuring Performance



ORBIT started from points in a 10x10 grid

Measuring Performance

Algorithms compared

- Ours (4 parallel workers)
- Direct (serial)
- pVTDirect (4 parallel workers)

- Since Ours involves a random sampling stream, each problem was duplicated 10 times.
- Each method evaluates the centroid first.

Measuring Performance

How would you measure how "efficiently" an algorithm finds multiple local minima?

Measuring Performance

Define

Let $\mathcal{X}^*$ be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*)|x^* \in \mathcal{X}^*\}$.
Let $x^*_{(i)}$ be the element of $\mathcal{X}^*$ corresponding to the value $f^*_{(i)}$.

Measuring Performance

Define

Let $\mathcal{X}^*$ be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*)|x^* \in \mathcal{X}^*\}$.
Let $x^*_{(i)}$ be the element of $\mathcal{X}^*$ corresponding to the value $f^*_{(i)}$.

How would you measure how "efficiently" an algorithm finds multiple local minima?

Measuring Performance

Define

Let $\mathcal{X}^*$ be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*) | x^* \in \mathcal{X}^*\}$.
Let $x^*_{(i)}$ be the element of $\mathcal{X}^*$ corresponding to the value $f^*_{(i)}$.

A method has found the global minimum within a level $\tau > 0$ at batch iteration k if it has found a point $\hat{x}$ satisfying:

$$f(\hat{x}) - f^*_{(1)} \leq (1 - \tau) \left(f(x_0) - f^*_{(1)} \right),$$

where x_0 is the starting point for the problem.

Measuring Performance

Define

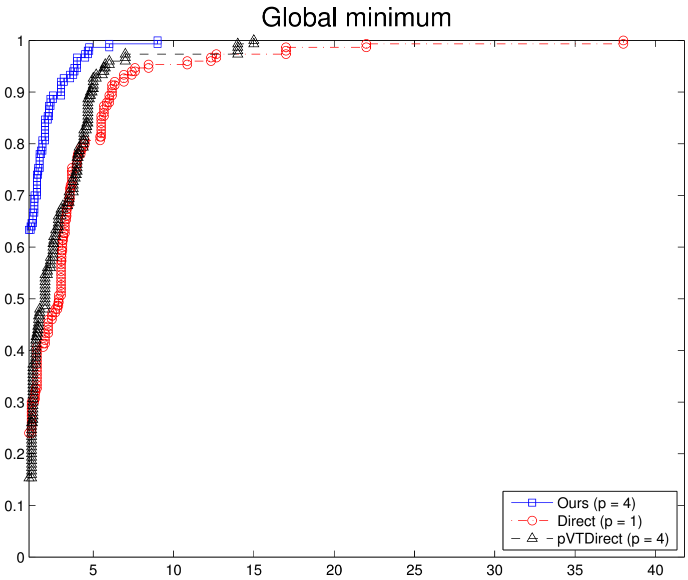
Let $\mathcal{X}^*$ be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*)|x^* \in \mathcal{X}^*\}$.
Let $x^*_{(i)}$ be the element of $\mathcal{X}^*$ corresponding to the value $f^*_{(i)}$.

A method has found the j best local minima within a level $\tau > 0$ at batch iteration k if:

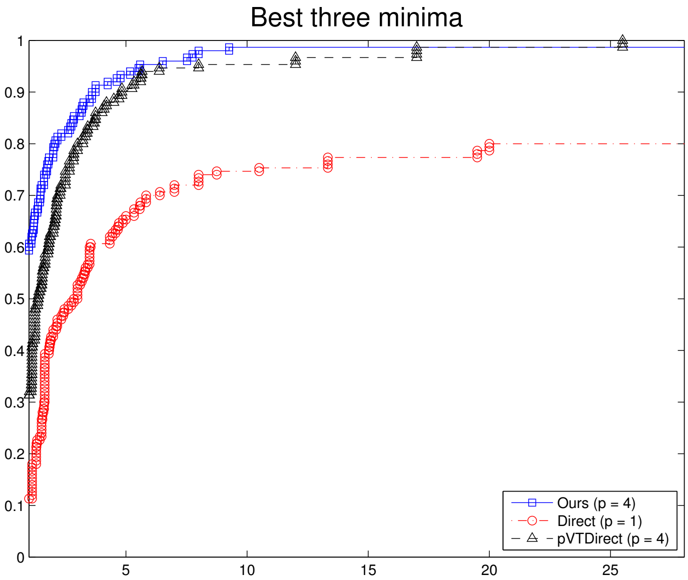
$$\left|\left\{x^*_{(1)}, \ldots, x^*_{(\bar{j})}\right\} \bigcap \left\{x^*_{(i)} | \exists \hat{x} \text{ s.t. } \left\|\hat{x} - x_{(i)}\right\| \leq \tau\right\}\right| \geq j$$

where $\bar{j}$ is the largest integer such that $f^*_{(\bar{j})} = f^*_{(j)}$.

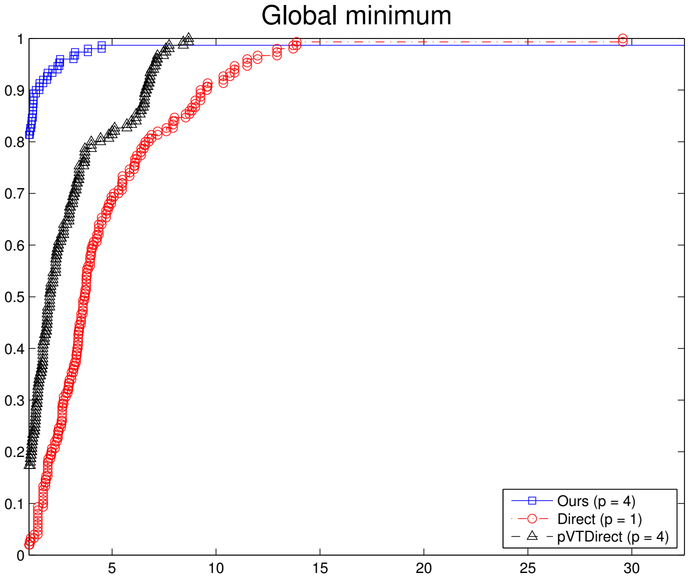
Measuring Performance: $\tau = 10^{-1}$



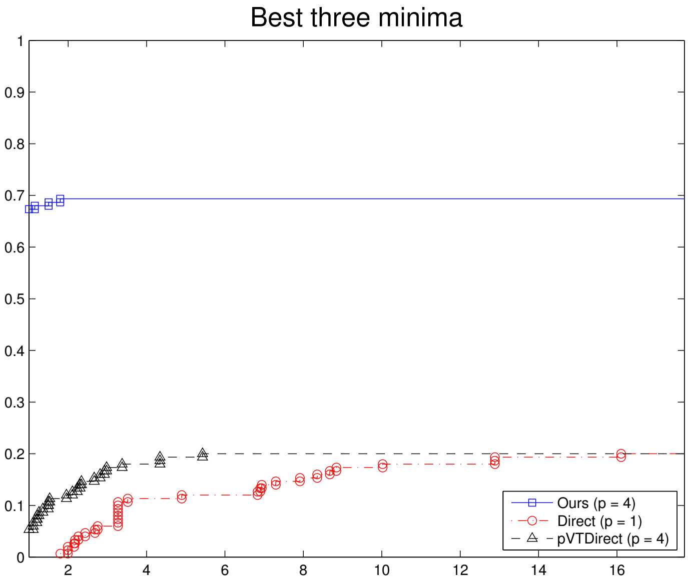
Measuring Performance: $\tau = 10^{-1}$



Measuring Performance: $\tau = 10^{-3}$



Measuring Performance: $\tau = 10^{-3}$



Conclusions and Future Work

Take Away:

- Derivative-free optimization is an active and applicable area of mathematics
- Local models efficiently use previous function evaluations
- Concurrent function evaluations can be used to help an algorithm find multiple local optima

Conclusions and Future Work

Take Away:

- Derivative-free optimization is an active and applicable area of mathematics
- Local models efficiently use previous function evaluations
- Concurrent function evaluations can be used to help an algorithm find multiple local optima

Working on:

- Industrial implementation of Our algorithm
- Accelerator design problems
- Developing rules for pausing local optimization runs

For students:

Computational math includes

- Applications
- Computer Programming
- Theory

For students:

Computational math includes

- Applications
- Computer Programming
- Theory

DFO is broad

- Geometry of evaluated points
- Analysis
- Stochastic/probabilistic functions
- Stochastic/probabilistic models
- Removing old points?

For students:

Computational math includes

- Applications
- Computer Programming
- Theory

DFO is broad

- Geometry of evaluated points
- Analysis
- Stochastic/probabilistic functions
- Stochastic/probabilistic models
- Removing old points?

Questions?

Other interests

- Sport Scheduling (Elitserien)
- Heavy-duty Vehicle Platooning
- Mathematics Outreach